BAYESIAN METHODS FOR REAL-TIME PITCH TRACKING

by

Umut Şimşekli

B.S., Computer Science and Engineering, Sabancı University, 2008

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2010

To my family ...

ACKNOWLEDGEMENTS

I would like to thank to my supervisor Assist. Prof. Ali Taylan Cemgil for all his support throughout the course of this research. It has been a great pleasure for me to have an academic advisor, a music instructor, and a friend at the same time. I also want to thank to my examiners, Prof. Ethem Alpaydın and Assoc. Prof. Muhittin Mungan for the valuable feedback they have provided for this thesis.

I also want to thank to the members of the Perceptual Intelligence Laboratory for their help, support and friendship. Particularly, many thanks go to Dr. Mehmet Gönen for his practical solutions for any kind of problems. It is not possible to mention everyone here, but I would like to thank Prof. Lale Akarun, Prof. Fikret Gürgen, my music instructors Sabri Tuluğ Tırpan, Volkan Hürsever, and finally Şanver Narin and the members of Defne Bilgi İşlem.

I would like to thank to my parents and my sister for everything I have and I succeeded. I tried so hard but I could not find any words to tell your meaning to me.

This thesis has been supported by the M.Sc. scholarship (2228) from the Scientific and Technological Research Council of Turkey (TÜBİTAK).

ABSTRACT

BAYESIAN METHODS FOR REAL-TIME PITCH TRACKING

In this thesis, we deal with probabilistic methods to track the pitch of a musical instrument in real-time. Here, we take the pitch as a physical attribute of a musical sound which is closely related to the frequency structure of the sound.

Pitch tracking is the task where we try to detect the pitch of a note in an online fashion. Our motivation was to develop an accurate and low-latency monophonic pitch tracking method which would be quite useful for the musicians who play low-pitched instruments. However, since accuracy and latency are conflicting quantities, simultaneously maximizing the accuracy and minimizing the latency is a hard task.

In this study, we propose and compare two probabilistic models for online pitch tracking: Hidden Markov Model (HMM) and Change Point Model (CPM). As opposed to the past research which has mainly focused on developing generic, instrument-independent pitch tracking methods, our models are instrument-specific and can be optimized to fit a certain musical instrument.

In our models, it is presumed that each note has a certain characteristic spectral shape which we call the *spectral template*. The generative models are constructed in such a way that each time slice of the audio spectra is generated from one of these spectral templates multiplied by a volume factor. From this point of view, we treat the pitch tracking problem as a template matching problem where the aim is to infer the active template and its volume as we observe the audio data.

In the HMM, we assume that the pitch labels have a certain temporal structure in such a way that the current pitch label depends on the previous pitch label. The volume variables are independent in time, which is not the natural case in terms of musical audio. In this model, the inference scheme is standard, straightforward, and fast.

In the CPM, we also introduce a temporal structure for the volume variables. In this way, the CPM enables explicit modeling of the damping structure of an instrument. As a trade off, the inference scheme of the CPM is much more complex than the HMM. After some degree, exact inference becomes impractical. For this reason, we developed an approximate inference scheme for this model.

The main goal of this work is to investigate the trade off in between latency and accuracy of the pitch tracking system. We conducted several experiments on an implementation which was developed in C++. We evaluated the performance of our models by computing the most-likely paths that were obtained via filtering or fixed-lag smoothing distributions. The evaluation was held on monophonic bass guitar and tuba recordings with respect to four evaluation metrics. We also compared the results with a standard monophonic pitch tracking algorithm (YIN). Both HMM and the CPM performed better than the YIN algorithm. The highest accuracy was obtained from the CPM, whereas the HMM was the fastest in terms of running time.

ÖZET

GERÇEK ZAMANLI NOTA TAKİBİ İÇİN BAYESÇİ YÖNTEMLER

Bu tezde notaların gerçek zamanlı perde takibi için Bayesçi yöntemler ele alınmıştır. Burada ses perdesini, sesin frekans yapısıyla yakından ilgili, fiziksel bir öznitelik olarak ele alıyoruz.

Nota takibi, bir notanın perdesinin çevrimiçi bir şekilde belirlenmesi görevidir. Motivasyonumuz, kalın sesli müzik aleti çalan müzisyenler için faydalı olabilecek, hassas ve düşük gecikmeli bir nota takip sistemi geliştirmekti. Ancak hassaslık ve gecikme çelişen iki nicelik olduğu için aynı anda hassiyeti enbüyütmek ve gecikmeyi enküçültmek zor bir görevdir.

Bu çalışmada, çevrimiçi nota takibi için iki olaslıksal model öneriyoruz: Saklı Markov Modeli (SMM) ve Değişim Noktası Modeli (DNM). Bu alanda yapılan önceki çalışmalarda genel, müzik aletine bağlı olmayan nota takip yöntemlerine odaklanılmıştı. Bunun aksine, bizim modellerimiz herhangi bir müzik aletine göre özelleştirilebilir ve belirli bir enstrumana göre eniyilenebilir.

Modellerimizde, her notanın spektral şablon adını verdiğimiz bir spektral yapıya sahip olduğunu varsayıyoruz. Üretici modellerimizi, ses spektrumunun bir zaman diliminin bu şablonlardan birinin bir gürlük katsayısıyla çarpılarak oluştuğu varsayımıyla kurduk. Bu açıdan, nota takibi problemini bir çeşit şablon eşleştirme problemi olarak ele alıyoruz. Amacımız, ses verisini gözlemledikçe hangi şablonun etkin olduğu ve şablona ait gürlük katsayısının ne olduğu çıkarımını yapabilmek.

SMM'de, notaların bir önceki notaya bağımlı olduğu bir zamansal yapıya sahip olduğunu varsayıyoruz. Gürlük değişkenini zamandan bağımsız ele alıyoruz. Ancak, müzik seslerini göz önünde bulundurunca bu varsayım doğal olmuyor. Diğer bir yandan, bu modellerde çıkarım yapmak için standart ve hızlı yöntemleri kullanabiliyoruz.

DNM'de, gürlük değişkenleri için de bir zamansal yapı öneriyoruz. Bu şekilde, DNM ile bir müzik aletinin sönümlenme yapısını açık şekilde modelleyebiliyoruz. Ancak ödünleşim sonucu, bu modelde çıkarım yapmak için çok daha karmaşık çıkarım yöntemleri kullanmamız gerekiyor. Ayrıca, bir noktadan sonra gerçek çıkarım uygulanamaz oluyor. Bu yüzden bu model için yaklaşık bir çıkarım şeması geliştirdik.

Bu çalışmanın temel hedefi, nota takip sisteminindeki gecikme ve hassasiyet arasındaki ödünleşimi incelemektir. Deneylerimizi C++ dilinde geliştirdiğimiz bir uygulama üzerinden yaptık. Modellerin başarılarını, süzgeçleme ve sabit gecikmeli düzleştirme dağılımlarından elde ettiğimiz en muhtemel yolları kullanarak hesapladık. Değerlendirmeyi tek sesli bas gitar ve tuba kayıtları üzerinde ve dört farklı ölçüt kullanarak yaptık. Ayrıca sonuçlarımızı standart bir perde takip algoritması olan YIN ile karşılaştırdık. İki modelimizle de YIN'den daha başarılı sonuçlar elde ettik. En yüksek hassasiyeti DNM, en yüksek hesaplama hızını ise SMM ile elde ettik.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

$\langle\cdot\rangle$	Expectation
$[\cdot]$	Returns 1 if the argument is true, returns 0 otherwise
a_{ij}	State transition probability in the HMM
$a_{ij}^{(\{0,1\})}$	State transition probability in the CPM
$\mathcal{B}[\cdot]$	Lower-bound of the likelihood
$\mathcal{BE}(\cdot)$	Bernoulli Distribution
c_τ	Change point indicator variable at time τ
D_r	Domain of variable r
D_v	Domain of variable v
f_0	Fundamental frequency
F	Number of frequency components
$\mathcal{G}(\cdot,\cdot)$	Gamma Distribution
$\mathcal{H}[\cdot]$	Entropy
I	Number of spectral templates
N	Number of Gamma potential that will be kept during pruning
$\mathrm{pa}(\cdot)$	Parent nodes of the parameter in a given graphical model
$\mathcal{PO}(\cdot)$	Poisson Distribution
Q	Objective in the EM derivations for the CPM
r_τ	Pitch indicator variable at time τ
$t_{\nu,i}$	Spectral template with pitch index i and frequency index ν
T	Number of time slices
v_τ	Volume variable at time τ
x_τ	Hidden variable at time τ
$x_{\nu,\tau}$	Audio spectrum with time index τ and frequency index ν
y_τ	Observed variable at time τ
α	Forward message
β	Backward message
γ	Gamma potential
$\Gamma(\cdot)$	Gamma function

$\delta(\cdot)$	Kronecker delta function
$\theta(\cdot)$	Damping function in the CPM
$\kappa(\cdot)$	Returns the normalization constant of a Gamma potential
Λ	Objective in the generic EM derivations
ν	Frequency index
τ	Time index

CPM	Change Point Model
HMM	Hidden Markov Model
MAP	Maximum a-posteriori
MIDI	Musical Instrument Digital Interface
NMF	Non-negative Matrix Factorization

1. INTRODUCTION

Computer music is the term that defines a multidisciplinary research field which aims to understand, analyze and synthesize music by incorporating the artistic and scientific information which is gained from computer science, electrical engineering, psychoacoustics, musicology, and music composition.

Among many subfields of computer music (such as music information retrieval, musical sound generation, algorithmic composing, etc.), *interactive computer music systems* became popular along with the rapid increase in the computational power. A computer music system is called to be interactive if it has the capacity to interact with musicians like a real musician. This requires efficient methods in order to response in real-time and comprehensive analysis and interpretation of music such as pitch, tempo, and rhythm analysis. An illustration of a pitch tracking based interactive computer music system is shown in Figure 1.1.

1.1. Levels of Music Representation

Music is represented in several ways. This representations have a certain hierarchy. On the highest level, there is the printed music (also known as sheet music). This representation contains all kinds of *high level* musical information, such as pitch, velocity, tempo, rhythm, vibrato, glissando, legato, accelerando, ritardando, and etc. This representation also allows the musicians to have their own interpretation of the music. Hence the music that is played by different musicians from the same sheet will not be the same. While going down at the hierarchy, there is the acoustic waveform representation on the lowest level. In this representation, we lose all the high level information which we had in the sheet representation. On the other hand, we always have the same output as opposed to the sheet representation.

Computers allow and require formal representations of music, where each detail of the representation is precisely specified (Dannenberg, 1993). Hence, in order to make

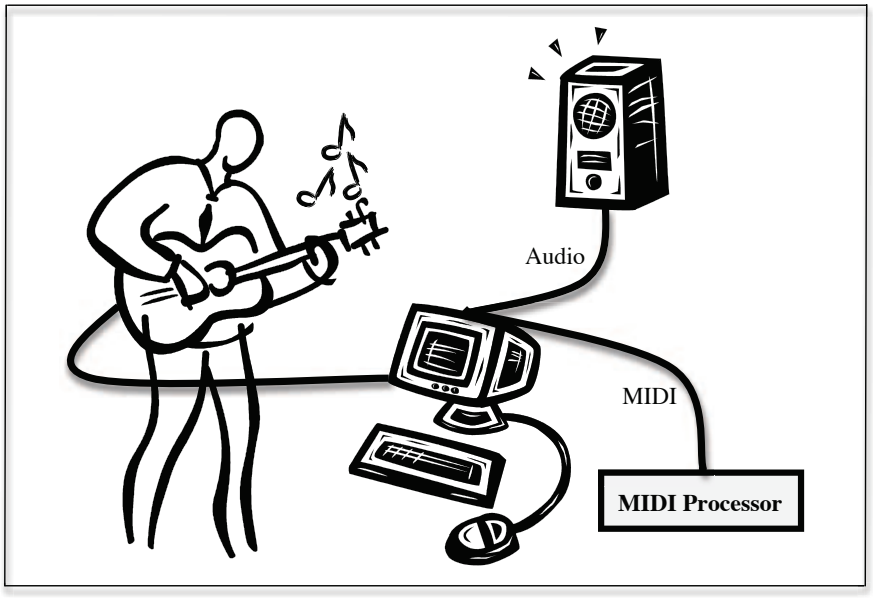
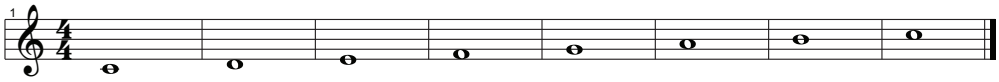


Figure 1.1. Illustration of an interactive computer music system. The audio signal is processed and converted to MIDI in real-time. The converted MIDI signals can be used for several purposes.

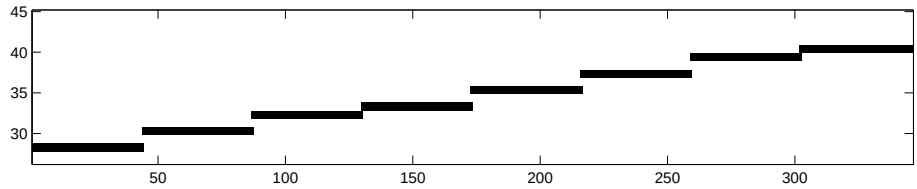
the high level music information available to computers, mid-level music representations, so called the symbolic representations, were developed.

MIDI (Musical Instrument Digital Interface) is the most common representation among the symbolic representations of music. The MIDI format was first developed in 1982 and it is the industry-standard symbolic music representation which enables to synchronize computers, synthesizers, keyboards, and MIDI controllers. MIDI signals do not convey any acoustic audio signal. Instead, they convey event messages which basically contain the information about pitch, start and end times of notes, and volume. This representation is similar to the sheet representation, while at the same time it is formal enough to satisfy computers' requirements. Figure 1.2 shows the different types of music representations. For further information about music representations, the reader is referred to (Dannenberg, 1993).

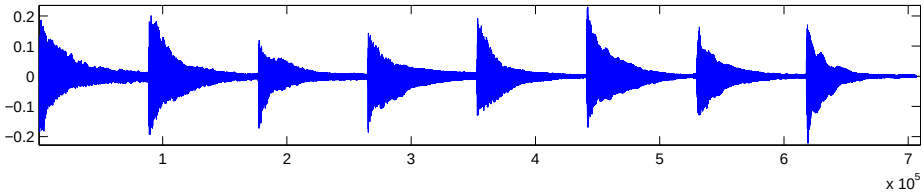
MIDI instruments are quite appropriate for interactive computer music applications since they do not need any acoustic processing. *Real-time pitch tracking* problem



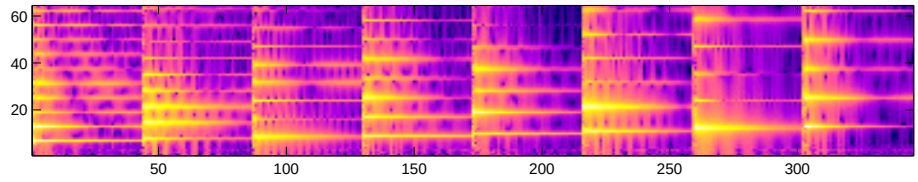
(a) Sheet representation of the C major scale



(b) The MIDI representation that corresponds to the sheet representation. Here we have the pitch information (on the y axis) and the note onset - offset information (on the x axis).



(c) The waveform representation of acoustic audio signal which was obtained from a piano.



(d) The log-spectra of the piano recording.

Figure 1.2. Different levels of the music representation.

appears when we want to use an acoustic instrument as a MIDI source. The goal of a real-time pitch tracking system can be seen as transforming the low-level audio signal to mid-level MIDI messages in real time. This requires real-time processing of the acoustic audio stream which is obtained as the musician plays the instrument.

1.2. Pitch Tracking

The term *pitch* is a psycho-acoustics term which is closely related to the frequency structure of a sound. It is one of the major properties of a musical sounds such as timbre and loudness. The pitch of a note determines the label of the note and how "high" or "low" the note is. For instance, in Figure 1.2(a), the first and last notes are both C (do). However the last one is one octave higher than the first one. Hence, their pitch labels are C4 and C5 respectively. Here C4 means the note C at the fourth octave.

Pitch is often referred as a subjective attribute of sound, not an objective physical quantity. However, in some contexts, it is used synonymously with the fundamental frequency (f_0) which is a physical quantity in fact. In this thesis we will not go into details of the properties of pitch and fundamental frequency. For more information about pitch and fundamental frequency, the reader is referred to (Christensen and Jakobsson, 2009).

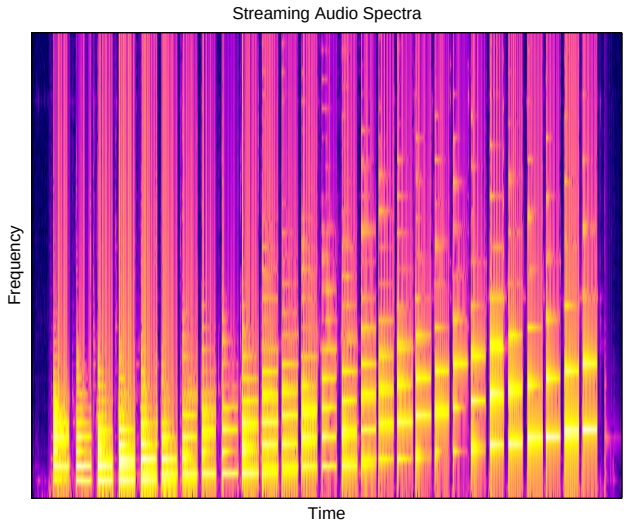
Pitch tracking is the task where we want to track the pitch labels while observing audio data. It is very similar to *object tracking* with many respects. In object tracking, the aim is to track an object's position and velocity while acquiring some time-series observations. Similar to this, in pitch tracking, what is tracked is also a temporal parameter. However, in this case, the parameter (or the quantity) is an attribute of musical sounds which is the *pitch*.

Pitch tracking is one of the most studied topics in the computer music field since it lies at the center of many applications. It is widely used in phonetics, speech coding, music information retrieval, music transcription, digital audio effects, and also

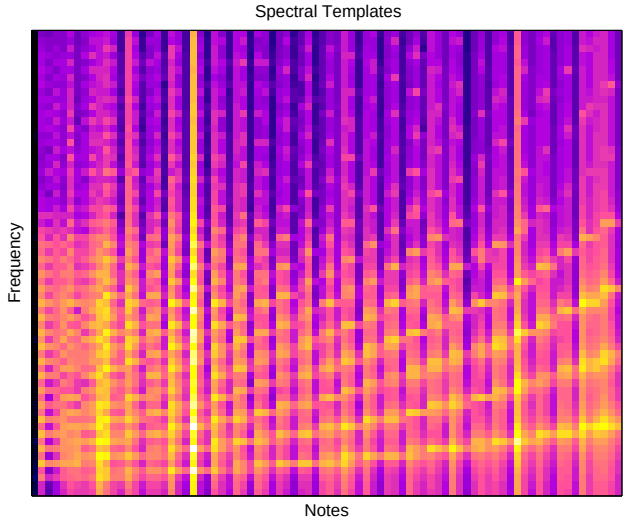
interactive musical performance systems. It is also used as a pre-processing step in more comprehensive music analysis applications such as chord recognition systems.

Many pitch tracking methods have been presented in the literature. Klapuri proposed an algorithmic approach for multipitch tracking in (Klapuri, 2008). Kashino et al. presented applied graphical models for polyphonic pitch tracking (Kashino *et al.*, 1998). Cemgil presented generative models for both monophonic and polyphonic pitch tracking (Cemgil, 2004). Orio et al. and Raphael proposed Hidden Markov Model based pitch tracking methods in (Orio and Sette, 2003) and (Raphael, 2002) respectively. On the other hand, using nonnegative matrix factorization (NMF) methods become popular at various audio processing applications. Different types of NMF models were proposed and tested on polyphonic music analysis, (Vincent *et al.*, 2008; Bertin *et al.*, 2009; Cont, 2006). There also exists practical commercial hardware devices such as Roland GI-20, Axon AX100 MKII, Axon AX50 USB, Yamaha G50, Sonuus G2M, Sonuus B2M, and etc. Most of these devices are designed to work with electric guitar and/or bass guitar, and they are also expensive as compared with the software products of pitch tracking even if they do not work perfectly.

In this study, we propose and compare two probabilistic models for online pitch tracking. Our aim is to convert the audio stream to a MIDI stream via a software program in such a way that the program would be as practical as the hardware devices which were mentioned above. As opposed to the past research which has mainly focused on developing generic, instrument-independent pitch tracking methods, our models are instrument-specific and can be optimized to fit a certain musical instrument. In our models, we represent the notes with *spectral templates* where a spectral template is a vector that captures the shape of a note's spectrum. Once we obtain the spectral templates of an instrument (via a training step), our system's goal becomes finding the note whose spectral template is more similar to the given audio spectra (see Figure 1.3). Thus, in that way, we can treat the pitch detection problem as a template matching problem.



(a) Spectral templates of a bass guitar.



(b) Audio spectra of a bass guitar recording.

Figure 1.3. Spectral templates and audio spectra which were obtained from a bass guitar. In (a), each column is a spectral template of a certain note and it captures the shape of the note's spectrum. After we obtain the spectral templates, the goal of our system becomes to determine which note's spectral template is most likely given this audio spectra in (b). It can also be observed that the spectral templates implicitly capture the harmonic structure of the signals.

Human auditory system has a complex structure and it can be obvious for a human to recognize the pitch of a sound quite accurately. However this not an easy task for a pitch tracking system. Possible difficulties for a pitch tracker mostly arise when polyphony, vibrato, and low pitches are introduced in a musical piece (Roads, 1996). Here, we mainly focus on monophonic pitch tracking of low pitched instruments even if our probabilistic models are extensible to polyphonic pitch tracking by using factorial models (Cemgil, 2006). The main concern of the work is reducing the pitch detection latency without compromising the detection quality. Here the term, latency is defined as the time difference between the true note onset and the time that the pitch tracker has computed its estimate. In our point of view, a pitch tracking method might have latency due to two reasons. The first reason is that the method cannot estimate the note accurately because it has not accumulated enough data yet. The second reason is the computational burden. With the increase of the computational power, the latter can be eliminated by using more powerful computers. Hence, in our work we will focus on decreasing the latency by increasing the accuracy at note onsets rather than trying to reduce the computational complexity. We tested our models on recordings of two low pitched instruments: tuba and bass guitar. This is challenging since estimating low pitches in shortest time is a difficult problem.

Apart from pitch tracking, template matching framework can also be used in various types of applications since we do not make any application-specific assumptions while constructing the models.

The rest of the thesis is organized as follows: in Chapter 2, we provide the necessary background information about the time series models. In Chapter 3, we present our pitch tracking models in detail. In Chapter 4 and 5, we describe the inference schemes and the training procedure. In Chapter 6, we present our experimental results. Finally, Chapter 7 concludes this thesis.

2. TIME-SERIES MODELS

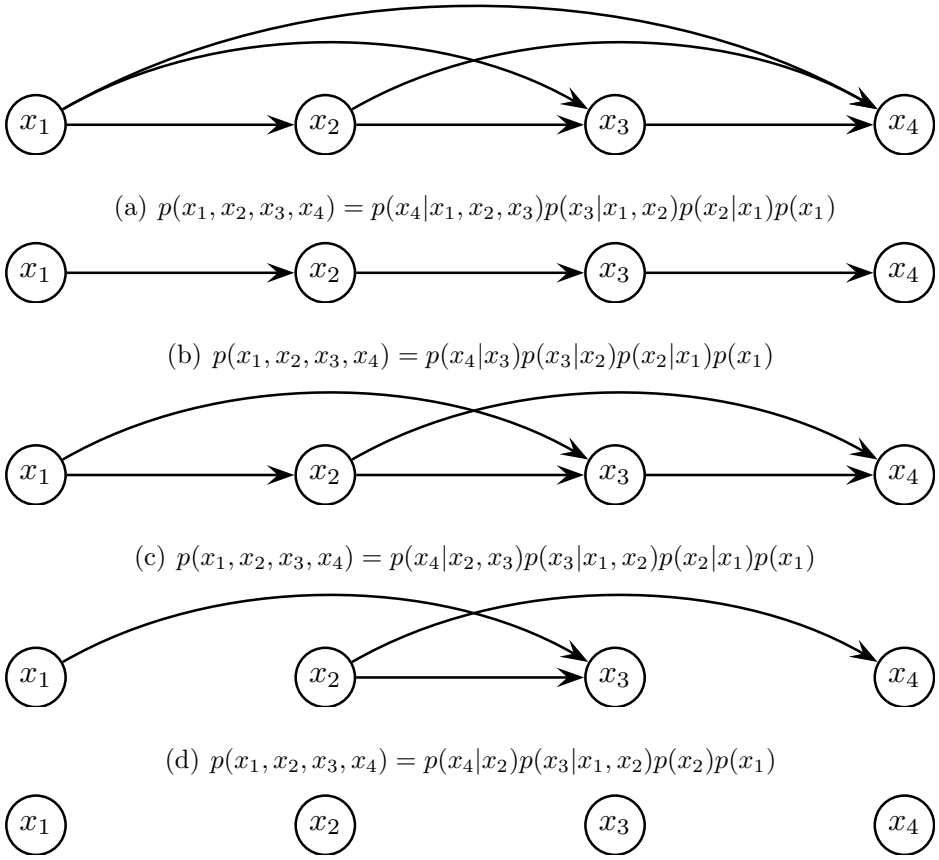
A time-series is a sequence of observations which are measured at a increasing set of time points (usually uniformly spaced). Since many problems can be defined in terms of time-series, time-series analysis has become very popular in various research areas including machine learning, acoustics, signal processing, image processing, mathematical finance, and econometrics (Excell *et al.*, 2007; West and Harrison, 1997; Godsill *et al.*, 2007; Harvey *et al.*, 2004; West and Harrison, 1997). Among many types of methods, Bayesian probabilistic models are quite natural for time-series analysis since they enable the use of many heuristics within a rigorous framework. Besides, they have shown convincing success in the field of computer music (Cemgil *et al.*, 2006; Virtanen *et al.*, 2008; Whiteley *et al.*, 2006; Whiteley *et al.*, 2007; Klapuri and Davy, 2006).

In a probabilistic model of a time-series $x_{1:T}$, the joint distribution of the observations, $p(x_{1:T})$ are specified [1] (Barber and Cemgil, 2010). In order the probabilistic model to be consistent with the causality of the time-series, we can utilize the chain rule and obtain the following recursion:

$$\begin{aligned} p(x_{1:T}) &= p(x_T|x_{1:T-1})p(x_{1:T-1}) \\ &= p(x_T|x_{1:T-1})p(x_{1:T-1}|x_{1:T-2})p(x_{1:T-2}) \\ &= p(x_T|x_{1:T-1})p(x_{1:T-1}|x_{1:T-2})p(x_{1:T-2}|x_{1:T-3})p(x_{1:T-3}) \\ &\vdots \\ &= \prod_{\tau=1}^{T} p(x_\tau|x_{1:\tau-1}), \end{aligned} \tag{2.1}$$

where $p(x_1|x_{1:0}) = p(x_1)$. This is a causal representation of the model where each variable depends on all past variables. However, in order the inference on a probabilistic model to be computationally tractable, different types of (in)dependence structures are assumed in different types of probabilistic models.

[1] Note that we use MATLAB's colon operator syntax in which $(1:T)$ is equivalent to $[1, 2, 3, ..., T]$ and $x_{1:T} = \{x_1, x_2, ..., x_T\}$.



(a) $p(x_1, x_2, x_3, x_4) = p(x_4|x_1, x_2, x_3)p(x_3|x_1, x_2)p(x_2|x_1)p(x_1)$

(b) $p(x_1, x_2, x_3, x_4) = p(x_4|x_3)p(x_3|x_2)p(x_2|x_1)p(x_1)$

(c) $p(x_1, x_2, x_3, x_4) = p(x_4|x_2, x_3)p(x_3|x_1, x_2)p(x_2|x_1)p(x_1)$

(d) $p(x_1, x_2, x_3, x_4) = p(x_4|x_2)p(x_3|x_1, x_2)p(x_2)p(x_1)$

(e) $p(x_1, x_2, x_3, x_4) = p(x_4)p(x_3)p(x_2)p(x_1)$

Figure 2.1. Graphical models with different conditional independence assumptions.

Graphical models provide an intuitive way to represent the conditional independence structure of a probabilistic model. We can rewrite the joint distribution by making use of a directed acyclic graph:

$$p(x_{1:T}) = \prod_{\tau=1}^{T} p(x_\tau|\mathrm{pa}(x_\tau)), \tag{2.2}$$

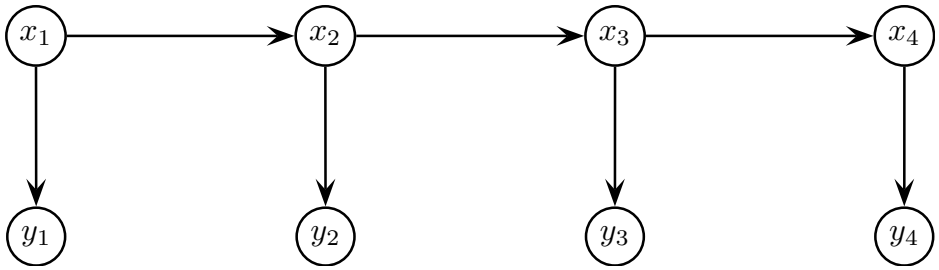


Figure 2.2. Graphical model of a Hidden Markov Model. x_τ represent the latent variables and y_τ represent the observations.

where $\text{pa}(x_\tau)$ denotes the *parent nodes* of x_τ. Figure 2.1 visualizes possible independence structures of a time-series. For further information on graphical models, the reader is referred to (Wainwright and Jordan, 2008; Parsons, 1998; Jordan, 2004).

2.1. Hidden Markov Model

A Hidden Markov Model (HMM) is a statistical model which is basically a Markov chain observed in noise. Here the underlying Markov chain is not observable, therefore hidden. What is observable in an HMM is also a stochastic process which is assumed to be generated from the hidden Markov chain (Cappé *et al.*, 2005). In this section we will represent the hidden variables with x_τ and the observed variables with y_τ. Conventionally, the underlying Markov chain, $x_{1:T}$ is called a state and in this study we will be dealing with discrete x_τ. Figure 2.2 shows the graphical model of a standard HMM.

As can be seen from the graphical model, the hidden state variable at time τ depends only on the state variable at time $\tau - 1$. This is called the Markov property:

$$p(x_\tau | x_{1:\tau-1}) = p(x_\tau | x_{\tau-1}). \tag{2.3}$$

Similarly, the observation at time τ depends only on the state variable at time τ,

$$p(y_\tau | y_{1:\tau-1}, x_{1:\tau}) = p(y_\tau | x_\tau). \tag{2.4}$$

In an HMM, the probability distribution in Equation (2.3) is called the *state transition model* and the distribution in Equation (2.4) is called the *observation model*. The HMM is called *homogeneous* if the state transition and the observation models do not depend on time index τ (Cappé *et al.*, 2005), which is our case in this study.

2.1.1. Example

As an example, we will consider a possible model for pitch labels in music. Let x_τ be pitch labels and y_τ be the discrete noisy observations of x_τ, where x_τ and y_τ have the same discrete domain $\mathcal{D}$, where $\mathcal{D} = \{\text{C,D,E,F,G,A,B}\}$. We can define the state transition model as follows:

$$p(x_\tau|x_{\tau-1}) = \begin{cases} p_0, & x_\tau = x_{\tau-1}, \\ p_1, & x_\tau \neq x_{\tau-1}. \end{cases} \tag{2.5}$$

Similarly, we can define the observation model:

$$p(y_\tau|x_\tau) = \begin{cases} q_0, & y_\tau = x_\tau, \\ q_1, & y_\tau \neq x_\tau. \end{cases} \tag{2.6}$$

Here $p_0 + p_1 = q_0 + q_1 = 1$. The assumption in the model is, at time τ, the pitch label will stay the same with p_0 probability or jump to another pitch with p_1 probability. We observe the true state with q_0 probability or a erroneous state with q_1 probability. Figure 2.3 shows synthetic data which are generated from this model.

Inference on the unobserved variables x_τ given the noisy observations y_τ, which is the main topic of interest, is computationally straight-forward for this model (Alpaydin, 2004). The conditional independence structure of the model allows us to derive generic recursions which will be covered in Chapter 4.

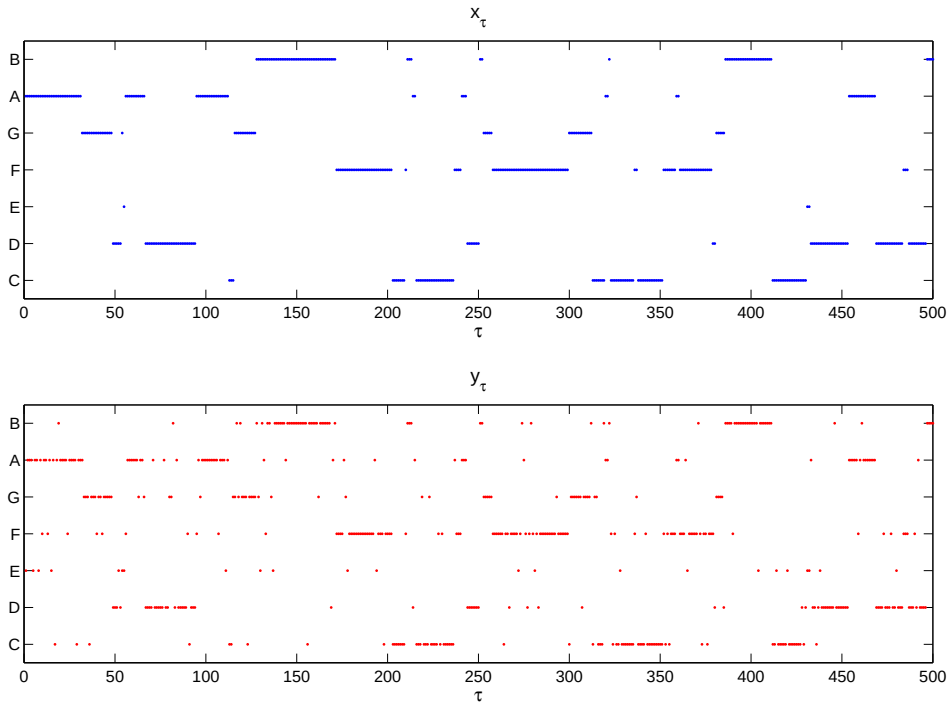


Figure 2.3. Synthetic data which are generated from the HMM. The upper plot can be viewed as a *piano-roll* representation of a musical piece. The lower plot corresponds to a noisy observation of the true states.

2.2. Change Point Model

In the classic time-series models, the underlying latent process is assumed to be either discrete (i.e. Hidden Markov Model) or continuous (i.e. Kalman Filter). These kinds of models have been shown to be successful in many problems from various research fields. However, in some cases selecting the underlying process either discrete or continuous would not be sufficient. Thanks to the increase in the computational power and the development in the state-of-the-art inference methods, we are able to construct more complex statistical models such as the change point models (Barber and Cemgil, 2010).

A change point model (CPM) is a switching state space model where the variables have a special structure. In a CPM, we have two latent variables: the **discrete** switch

variable c_τ and the **continuous** variable x_τ. While the switch variable is off ($c_\tau = 0$), x_τ follows the pre-defined structure that depends on $x_{\tau-1}$. On the other hand, at the time when the switch variable is on ($c_\tau = 1$), x_τ is reset to a new value independent from the previous values. A generic change point model can be defined as follows:

$$\begin{aligned} c_\tau | c_{\tau-1} &\sim p(c_\tau | c_{\tau-1}) \\ x_\tau | c_\tau, x_{\tau-1} &\sim \begin{cases} p_0(x_\tau | x_{\tau-1}), & c_\tau = 0 \\ p_1(x_\tau), & c_\tau = 1 \end{cases} \\ y_\tau | x_\tau &\sim p(y_\tau | x_\tau). \end{aligned} \tag{2.7}$$

In this model, the switch variables c_τ form a Markov chain. Besides, conditioned on c_τ, x_τ also form a Markov chain. The graphical model representation of a CPM is shown in Figure 2.4.

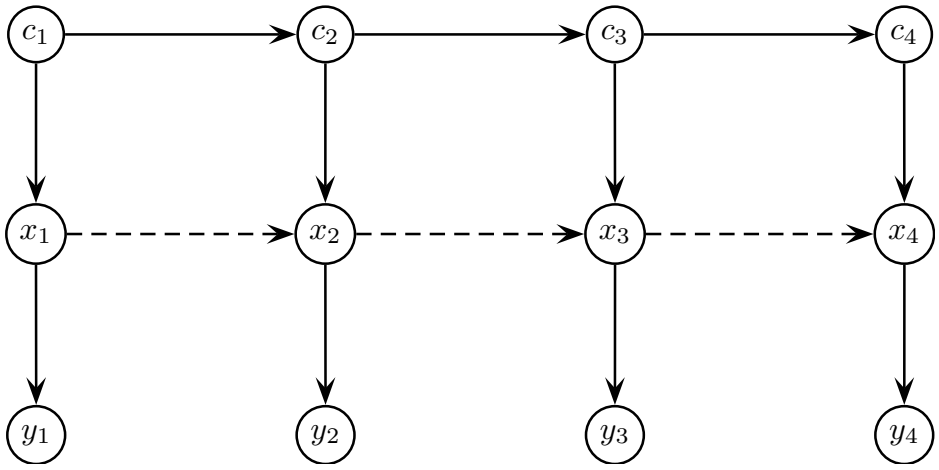


Figure 2.4. Graphical model of a Change Point Model. c_τ represent the binary switch variables. x_τ are the continuous latent variables. y_τ are the observations.

2.2.1. Example

The CPM is powerful at modeling the step changes in a continuous dynamical process. For instance, we can model note onsets and volume of a musical piece by

utilizing the CPM. Consider the following model:

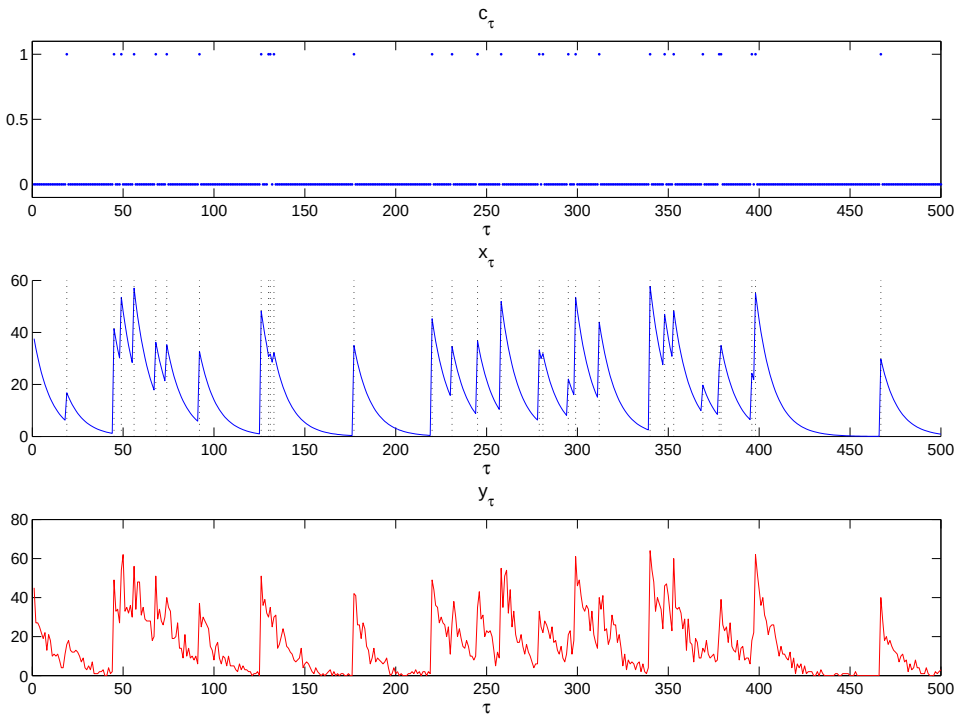
$$
\begin{aligned}
c_\tau &\sim \mathcal{BE}(c_\tau; w) \\
x_\tau | c_\tau, x_{\tau-1} &\sim \begin{cases} \delta(x_\tau - \theta x_{\tau-1}), & c_\tau = 0 \\ \mathcal{G}(x_\tau; a, b), & c_\tau = 1 \end{cases} \\
y_\tau | x_\tau &\sim \mathcal{PO}(y_\tau; x_\tau).
\end{aligned} \tag{2.8}
$$

Here $0 < \theta < 1$ and the symbols $\mathcal{BE}$, $\mathcal{G}$ and $\mathcal{PO}$ represent the Bernoulli, Gamma and the Poisson distributions respectively, where

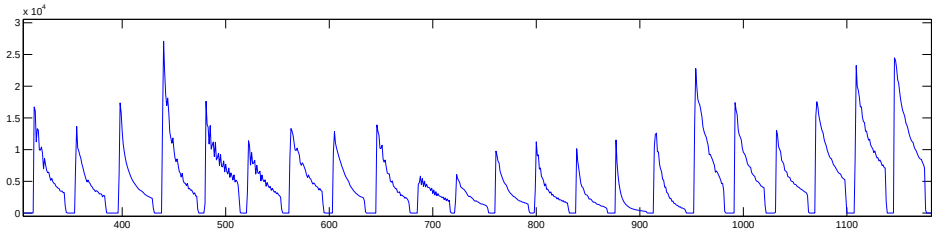
$$
\begin{aligned}
\mathcal{BE}(c; w) &= \exp(c \log w + (1 - c)(\log(1 - w)) \\
\mathcal{G}(x; a, b) &= \exp((a - 1) \log x - bx - \log \Gamma(a) + a \log(b)) \\
\mathcal{PO}(y; \lambda) &= \exp(y \log \lambda - \lambda - \log \Gamma(y + 1)).
\end{aligned} \tag{2.9}
$$

In this model, the switch variables c_τ determine the occurrence of note onsets and the continuous variable x_τ determine the instant volume of the given note without considering its label.

Compared to the Hidden Markov Model, making inference on the CPM is not straight-forward. The memory requirements of the inference scheme grow linearly with time and exact inference become intractable after some point. For this reason the inference scheme needs approximations which will be covered in Chapter 4.



(a) Synthetic data which are generated from the CPM.



(b) Spectral energy plot of a real bass guitar recording

Figure 2.5. Synthetic volume data and real volume data. Note that the synthetic data is very similar to the real data.

3. MONOPHONIC PITCH TRACKING

In this study, we would like to infer a predefined set of pitch labels from streaming audio data. Our approach to this problem is model based. We will construct two probabilistic generative models that relate a latent event label to the actual audio recording, in this case audio is represented by the magnitude spectrum. We define $x_{\nu,\tau}$ as the magnitude spectrum of the audio data with frequency index ν and time index τ, where $\tau \in \{1, 2, ..., T\}$ and $\nu \in \{1, 2, ..., F\}$.

For each time frame τ, we define an indicator variable r_τ on a discrete state space D_r, which determines the label we are interested in. In our case D_r consists of note labels such as {C4, C#4, D4, D#4, ..., C6}. The indicator variables r_τ are hidden since we do not observe them directly. For online processing, we are interested in the computation of the following posterior quantity, also known as the filtering density:

$$p(r_\tau | x_{1:F,1:\tau}). \tag{3.1}$$

Similarly, we can also compute the most likely label trajectory given all the observations

$$r^*_{1:T} = \underset{r_{1:T}}{\operatorname{argmax}}\, p(r_{1:T} | x_{1:F,1:T}). \tag{3.2}$$

This latter quantity requires that we accumulate all data and process in a batch fashion. There are also other quantities, called "fixed lag smoothers" that between those two extremes. For example, at time τ we can compute

$$p(r_\tau | x_{1:F,1:\tau+L}) \tag{3.3}$$

and

$$r^*_\tau = \underset{r_\tau}{\operatorname{argmax}}\, p(r_{1:\tau+L} | x_{1:F,1:\tau+L}), \tag{3.4}$$

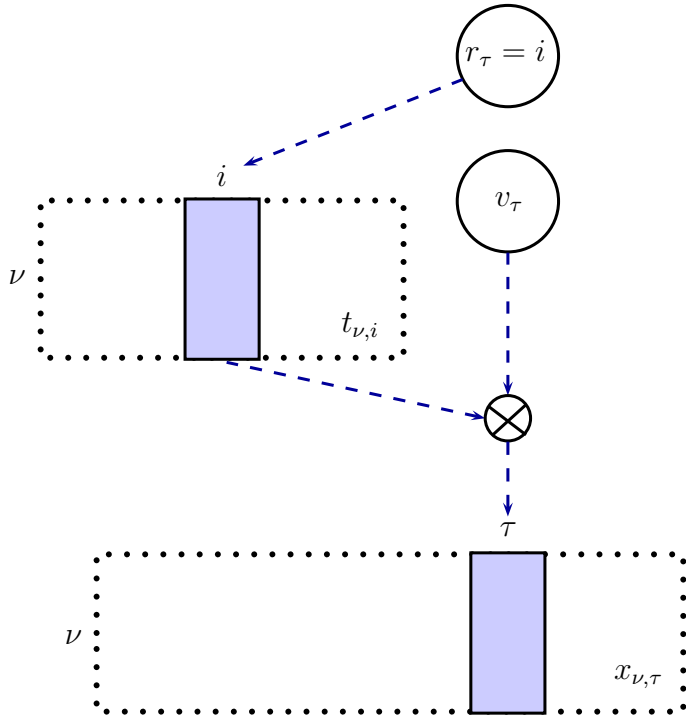


Figure 3.1. The block diagram of the probabilistic models. The indicator variables, r_τ choose which template to be used. The chosen template is multiplied by the volume parameter v_τ in order to obtain the magnitude spectrum, $x_{\nu,\tau}$.

where L is a specified lag and it determines the trade off between the accuracy and the latency. By accumulating a few observations from the future, the detection at a specific frame can be eventually improved by introducing a slight latency. Hence we have to fine-tune this parameter in order to have the optimum results.

3.1. Models

In our models, the main idea is that each event has a certain characteristic spectral shape which is rendered by a specific volume. The spectral shapes that we denote as *spectral templates* are denoted by $t_{\nu,i}$. The ν index is again the frequency index and the index i indicates the pitch labels. Here, i takes values between 1 and I, where I is the number of different spectral templates. The volume variables v_τ define the overall amplitude factor, by which the whole template is multiplied. An overall sketch of the model is given in Figure 3.1.

3.1.1. Hidden Markov Model

Hidden Markov Models have been widely studied in various types of applications such as audio processing, natural language processing, and bioinformatics. Like in many computer music applications, HMMs have also been used in pitch tracking applications (Orio and Sette, 2003; Raphael, 2002).

We define the probabilistic model as follows:

$$\begin{aligned} r_0 &\sim p(r_0) \\ r_\tau | r_{\tau-1} &\sim p(r_\tau | r_{\tau-1}) \\ v_\tau &\sim \mathcal{G}(v_\tau; a_v, b_v) \\ x_{\nu,\tau} | v_\tau, r_\tau &\sim \prod_{i=1}^{I} \mathcal{PO}(x_{\nu,\tau}; t_{\nu,i} v_\tau)^{[r_\tau = i]}. \end{aligned} \tag{3.5}$$

Here $[x] = 1$ if x is true, $[x] = 0$ otherwise.

In some recent work on polyphonic pitch tracking, Poisson observation model was used in the Bayesian non-negative matrix factorization models (NMF) (Cemgil, 2009). Since our probabilistic models are similar to NMF models, we choose the Poisson distribution as the observation model. We also choose Gamma prior on v_τ to preserve conjugacy and make use of the scaling property of Gamma distribution.

Moreover, we choose Markovian prior on the indicator variables, r_τ which means r_τ depends only on $r_{\tau-1}$. We use three states to represent a note: one state for the attack part, one for the sustain part, and one for the release part. We also use a single state in order to represent silence. Figure 3.2 shows the graphical model of the HMM, Figure 3.3 visualizes the different parts of a note, and Figure 3.4 shows the Markovian structure in more detail.

In this probabilistic model we can integrate out analytically the volume variables, v_τ. It is easy to check that once we do this, provided the templates $t_{\nu,i}$ are

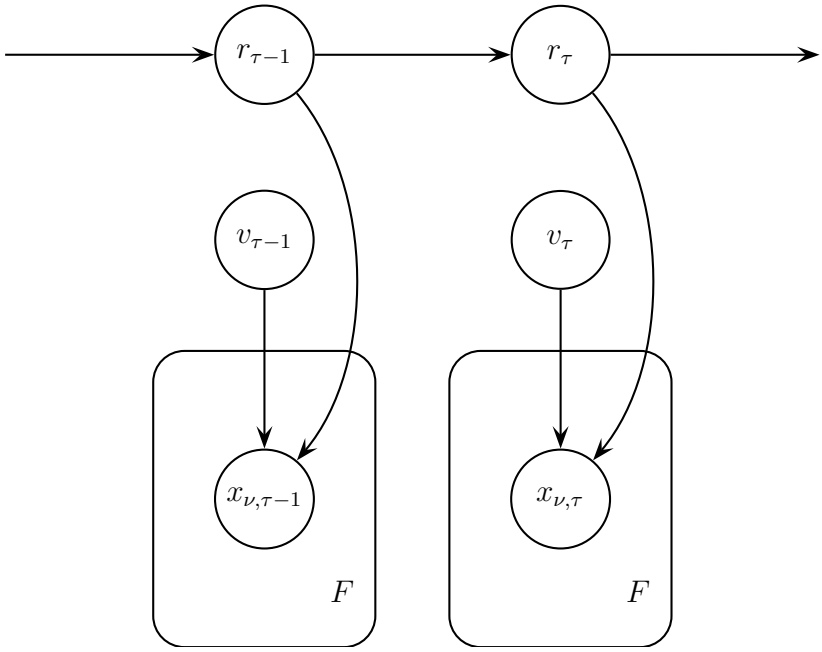


Figure 3.2. Graphical model of the HMM. The index ν takes values between 1 and F.

already known, the model reduces to a standard Hidden Markov Model (HMM) with a Compound Poisson observation model.

3.1.2. Change Point Model

In addition to the HMM, in the change point model (CPM), the volume parameter v_τ has a specific structure which depends on $v_{\tau-1}$ (i.e. staying constant, monotonically increasing or decreasing and etc.). But at certain unknown times, it jumps to a new value independently from $v_{\tau-1}$. We call these times as *"change points"* and the occurrence of a change point is determined by the switch variable c_τ. If c_τ is on, in other words if c_τ is equal to 1, then a change point has occurred at time τ.

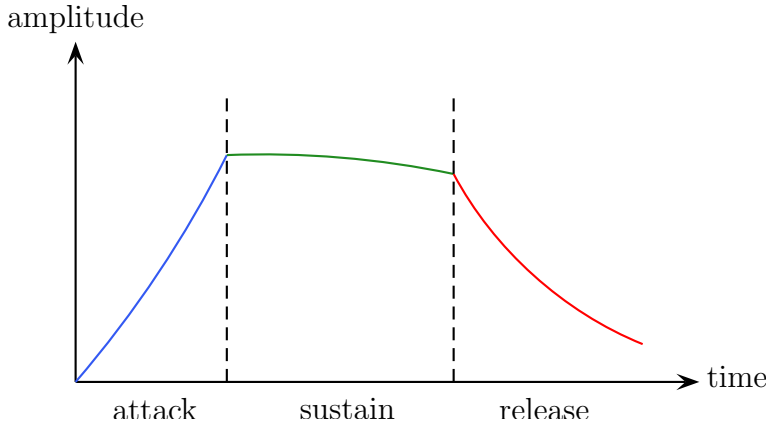


Figure 3.3. The structure of a note. The *attack* part of a note is usually a noise-like, non-stationary signal. In the *sustain* part, the signal attains its harmonic structure and the volume is pretty much constant. In the *release* part, the signal damps rapidly.

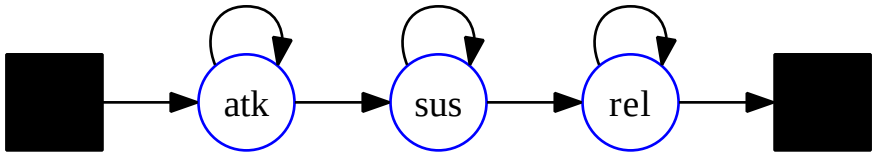


Figure 3.4. The state transition diagram of the indicator variable r_τ. Here *atk*, *sus*, and *rel* refers to the attack, sustain, and release parts of a note respectively. The first black square can be either the silence or a note release state. Similarly the second black square can be either a silence or a note attack state.

The formal definition of the generative model is given below:

$$
\begin{aligned}
v_0 &\sim \mathcal{G}(v_0; a_0, b_0) \\
r_0 &\sim p(r_0) \\
c_\tau &\sim \mathcal{BE}(c_\tau; w) \\
r_\tau | c_\tau, r_{\tau-1} &\sim \begin{cases} p_0(r_\tau | r_{\tau-1}), & c_\tau = 0 \\ p_1(r_\tau | r_{\tau-1}), & c_\tau = 1 \end{cases} \\
v_\tau | c_\tau, r_\tau, v_{\tau-1} &\sim \begin{cases} \delta(v_\tau - \theta(r_\tau) v_{\tau-1}), & c_\tau = 0 \\ \mathcal{G}(v_\tau; a_v, b_v), & c_\tau = 1 \end{cases} \\
x_{\nu,\tau} | v_\tau, r_\tau &\sim \prod_{i=1}^{I} \mathcal{PO}(x_{\nu,\tau}; t_{\nu,i} v_\tau)^{[r_\tau = i]}.
\end{aligned}
\tag{3.6}
$$

Here, $\delta(x)$ is the Kronecker delta function which is defined by $\delta(x) = 1$ when $x = 0$, and $\delta(x) = 0$ elsewhere. The graphical representation of the probabilistic model is given in Figure 3.5.

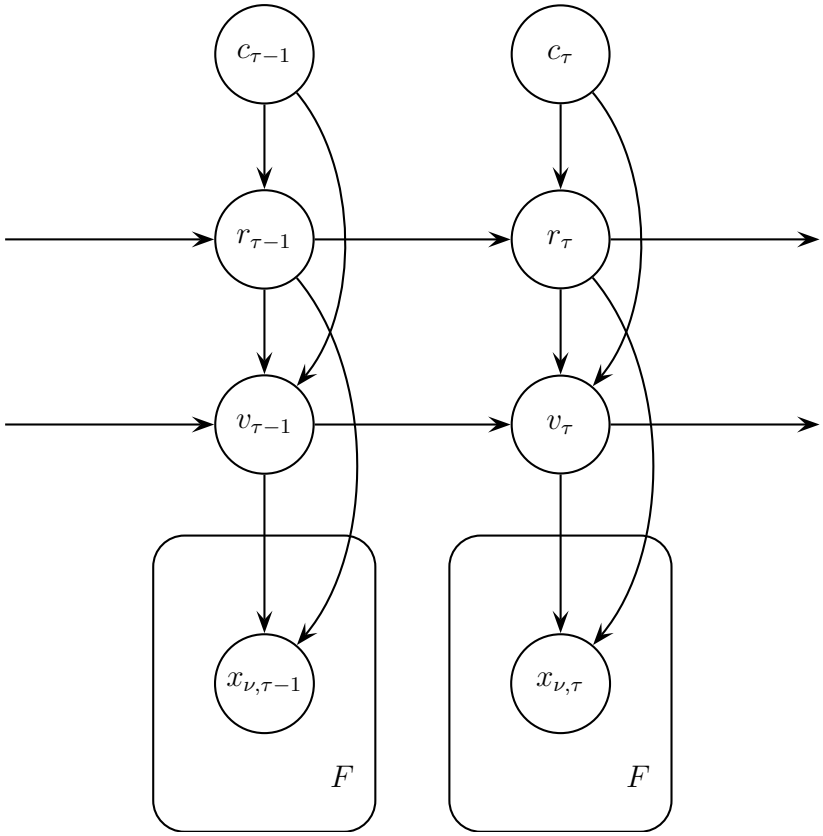


Figure 3.5. Graphical model of the CPM. The index ν takes values between 1 and F.

The $\theta(r_\tau)$ parameter determines the specific structure of the volume variables. Our selection of $\theta(r_\tau)$ is as follows:

$$\theta(r_\tau) = \begin{cases} \theta_1, & \text{if } r_\tau \text{ is attack,} \\ \theta_2, & \text{if } r_\tau \text{ is sustain,} \\ \theta_3, & \text{if } r_\tau \text{ is release.} \end{cases} \tag{3.7}$$

$\theta(r_\tau)$ gives flexibility to the CPM since we can adjust it with respect to the instrument whose sound would be processed (i.e. we can select $\theta(r_\tau) = 1$ for woodwind instruments by assuming the volume of a single note would stay approximately constant). Figure 3.6 visualizes example templates and synthetic data which are generated from the CPM.

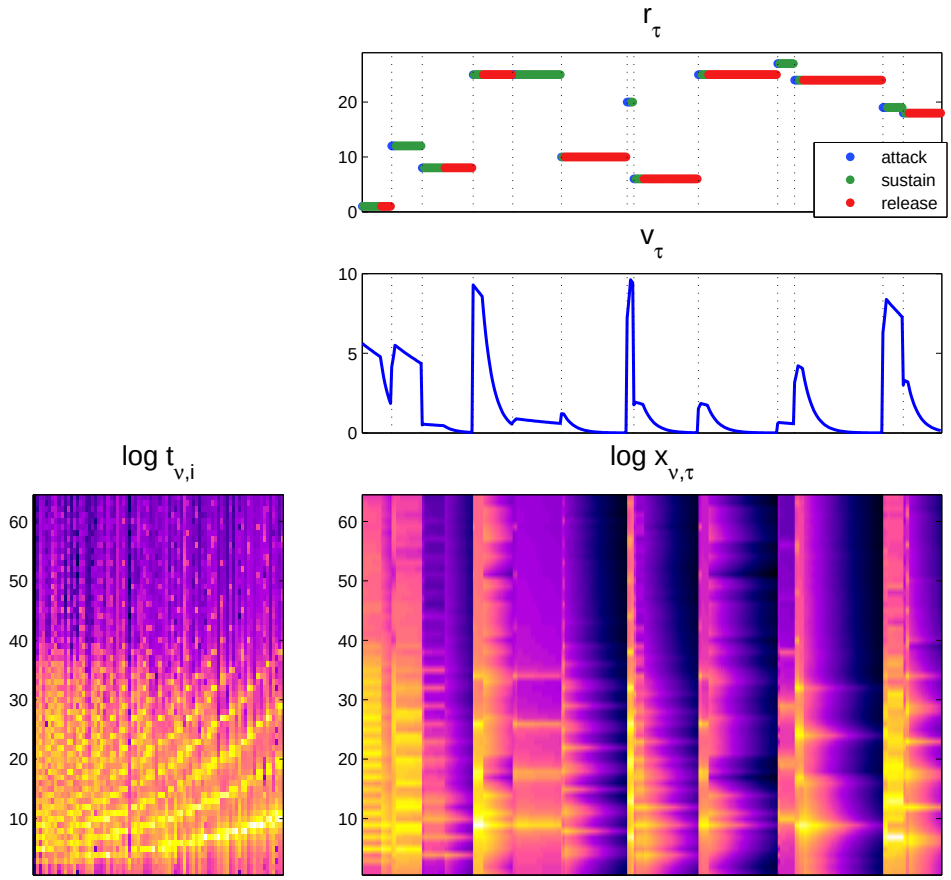


Figure 3.6. Spectral templates of a tuba and synthetic data generated from the CPM. The topmost right figure shows a realization of the indicator variables r_τ and the second topmost figure shows a realization of the volume variables v_τ. Here we set $\theta_{1:3} = \{1.10, 0.99, 0.90\}$. With this parametrization, we force the volume variables to increase during the attack parts, slowly damp at the sustain parts and rapidly damp during the release parts of the notes. The θ parameters should be determined by taking the audio structure into account (i.e. $\theta(r_\tau)$ should be different for higher sustained sounds, percussive sounds, woodwinds, etc.).

4. INFERENCE

Inference is a fundamental issue in probabilistic modeling where we ask the question "what can be the hidden variables as we have some observations?" (Cappé *et al.*, 2005). This chapter deals with the inference schemes of our two probabilistic models. We present the methods by which we can compute the filtering, smoothing, fixed-lag smoothing distributions; the Viterbi, and the fixed-lag Viterbi paths (see Equations (3.1), (3.2), and (3.3)) in detail.

4.1. Inference on the Hidden Markov Model

As we mentioned in Subsection 3.1.1, we can integrate out analytically the volume variables, v_τ. Hence, given that the $t_{\nu,i}$ are already known, the model reduces to a standard Hidden Markov Model (HMM) with a Compound Poisson observation model as shown below (see Appendix A.1 for details):

$$\begin{aligned} p(x_{1:F,1:\tau}|r_\tau = i) &= \int dv_\tau \, \exp\Big(\sum_{\nu=1}^{F} \log \mathcal{PO}(x_{\nu,\tau}; v_\tau t_{\nu,i}) + \log \mathcal{G}(v_\tau; a_v, b_v)\Big) \\ &= \frac{\Gamma\big(\sum_{\nu=1}^{F} x_{\nu,\tau} + a\big)}{\Gamma(a) \prod_{\nu=1}^{F} \Gamma(x_{\nu,\tau}+1)} \frac{b^a \prod_{\nu=1}^{F} t_{\nu,i}^{x_{\nu,\tau}}}{\big(\sum_{\nu=1}^{F} t_{\nu,i} + b\big)^{\sum_{\nu=1}^{F} x_{\nu,\tau}+a}}. \end{aligned} \tag{4.1}$$

Since we have a standard HMM from now on, the inference of the latent indicator variables r_τ given the noisy observations $x_{\nu,\tau}$ becomes straight-forward. We can compute the *filtering distribution* $p(r_\tau|x_{1:F,\tau})$ by first obtaining the joint distribution $p(r_\tau, x_{1:F,1:\tau})$. Considering the conditional independence assumptions of the HMM, we

can obtain the following recursion:

$$\begin{aligned} p(r_\tau|x_{1:F,1:\tau}) &\propto p(r_\tau, x_{1:F,1:\tau}) \\ &= \sum_{r_{\tau-1}} p(r_\tau, r_{\tau-1}, x_{1:F,1:\tau-1}, x_{1:F,\tau}) \\ &= \sum_{r_{\tau-1}} p(x_{1:F,\tau}|r_\tau, \cancel{r_{\tau-1}, x_{1:F,1:\tau-1}}) p(r_\tau|r_{\tau-1}, \cancel{x_{1:F,1:\tau-1}}) p(r_{\tau-1}, x_{1:F,1:\tau-1}) \\ &= p(x_{1:F,\tau}|r_\tau) \sum_{r_{\tau-1}} p(r_\tau|r_{\tau-1}) p(r_{\tau-1}, x_{1:F,1:\tau-1}). \end{aligned} \tag{4.2}$$

This recursion yields to the well-known forward algorithm. We can define the forward messages as follows:

$$\alpha_{\tau|\tau-1}(r_\tau) = p(r_\tau, x_{1:F,1:\tau-1}) \tag{4.3}$$

$$\alpha_{\tau|\tau}(r_\tau) = p(r_\tau, x_{1:F,1:\tau}). \tag{4.4}$$

By making use of these variables, we obtain the following recursions:

$$\alpha_{\tau|\tau-1}(r_\tau) = \sum_{r_{\tau-1}} p(r_\tau|r_{\tau-1}) \alpha_{\tau-1|\tau-1}(r_{\tau-1}) \tag{4.5}$$

$$\alpha_{\tau|\tau}(r_\tau) = p(x_{1:F,\tau}) \alpha_{\tau|\tau-1}(r_\tau). \tag{4.6}$$

Here Equation (4.5) and Equation (4.6) are also known as the *prediction step* and the *update step* respectively. Similar to the forward messages, the backward messages are defined as follows:

$$\beta_{\tau|\tau+1}(r_\tau) = p(x_{1:F,\tau+1:T}|r_\tau) \tag{4.7}$$

$$\beta_{\tau|\tau}(r_\tau) = p(x_{1:F,\tau:T}|r_\tau). \tag{4.8}$$

We also define the backward recursions:

$$\beta_{\tau|\tau+1}(r_\tau) = \sum_{r_{\tau+1}} p(r_{\tau+1}|r_\tau)\beta_{\tau+1|\tau+1}(r_{\tau+1}) \tag{4.9}$$

$$\beta_{\tau|\tau}(r_\tau) = p(x_{1:F,\tau}|r_\tau)\beta_{\tau|\tau+1}(r_\tau). \tag{4.10}$$

By making use of the contributions from the past and future, we obtain the *smoothing distribution* $p(r_\tau|x_{1:F,1:T})$:

$$\begin{aligned} p(r_\tau|x_{1:F,1:T}) &= \alpha_{\tau|\tau-1}(r_\tau)\beta_{\tau|\tau}(r_\tau) \\ &= \alpha_{\tau|\tau}(r_\tau)\beta_{\tau|\tau+1}(r_\tau). \end{aligned} \tag{4.11}$$

The Viterbi path is also obtained by replacing the *summations* over r_τ by *maximization* in the forward recursion. Hence, the most probable state sequence is computed as:

$$\begin{aligned} r^*_{1:T} &= \operatorname*{argmax}_{r_{1:T}} p(r_{1:T}|x_{1:F,1:T}) \\ &= \operatorname*{argmax}_{r_T}(x_{1:F,T}|r_T) \operatorname*{argmax}_{r_{T-1}} p(r_T|r_{T-1}) \ldots \operatorname*{argmax}_{r_2} p(r_3|r_2)p(x_{1:F,2}|r_2) \\ &\quad \operatorname*{argmax}_{r_1} p(r_2|r_1)p(x_{1:F,1}|r_1)p(r_1) \end{aligned} \tag{4.12}$$

which is equivalent to dynamic programming.

4.2. Inference on the Change Point Model

While making inference on the CPM, our task is finding the posterior probability of the indicator variables, r_τ and volume variables v_τ. If the state space of v_τ, D_v was discrete, then the CPM would reduce to an ordinary HMM on $D_r \times D_v$. However when D_v is continuous, which is our case, an exact forward backward algorithm cannot be implemented in general. This is due to the fact that the prediction density $p(r_\tau, v_\tau|x_{1:F,\tau})$ needs to be computed by integrating over $v_{\tau-1}$ and summing over $r_{\tau-1}$.

The summation over $r_{\tau-1}$ renders the prediction density a mixture model where the number of mixture component grow exponentially with τ. In this section we will describe the implementation of exact forward backward algorithm for the CPM and the pruning technique that we use for real-time applications.

The forward backward algorithm is a well known algorithm for computing the marginals of form $p(r_\tau, v_\tau|x_{1:F,\tau})$. We define the following forward messages:

$$\alpha_{0|0}(r_0, v_0) = p(r_0, v_0) \tag{4.13}$$

$$\alpha_{\tau|\tau-1}(c_\tau, r_\tau, v_\tau) = p(c_\tau, r_\tau, v_\tau, x_{1:F,1:\tau-1}) \tag{4.14}$$

$$\alpha_{\tau|\tau}(c_\tau, r_\tau, v_\tau) = p(c_\tau, r_\tau, v_\tau, x_{1:F,1:\tau}) \tag{4.15}$$

where $\tau \in \{1, 2, ..., T\}$. These messages can be computed by the following recursion:

$$\begin{aligned} \alpha_{\tau|\tau-1}(c_\tau, r_\tau, v_\tau) &= \sum_{c_{\tau-1}} \sum_{r_{\tau-1}} \int dv_{\tau-1}\, p(c_\tau, r_\tau, v_\tau|r_{\tau-1}, v_{\tau-1}) \\ &\quad \alpha_{\tau-1|\tau-1}(c_{\tau-1}, r_{\tau-1}, v_{\tau-1}) \qquad (4.16) \\ \alpha_{\tau|\tau}(c_\tau, r_\tau, v_\tau) &= p(x_{1:F,\tau}|c_\tau, r_\tau, v_\tau)\alpha_{\tau|\tau-1}(c_\tau, r_\tau, v_\tau) \\ &= p(x_{1:F,\tau}|r_\tau, v_\tau)\alpha_{\tau|\tau-1}(c_\tau, r_\tau, v_\tau). \qquad (4.17) \end{aligned}$$

We also define the backward messages and recursions similarly:

$$\begin{aligned} \beta_{T|T}(c_T, r_T, v_T) &= p(x_{1:F,T}|c_T, r_T, v_T) \qquad (4.18) \\ \beta_{\tau|\tau+1}(c_\tau, r_\tau, v_\tau) &= p(x_{1:F,\tau+1:T}|c_\tau, r_\tau, v_\tau) \\ &= \sum_{c_{\tau+1}} \sum_{r_{\tau+1}} \int dv_{\tau+1}\, p(c_{\tau+1}, r_{\tau+1}, v_{\tau+1}|r_\tau, v_\tau) \\ &\quad \beta_{\tau+1|\tau+1}(c_{\tau+1}, r_{\tau+1}, v_{\tau+1}) \qquad (4.19) \\ \beta_{\tau|\tau}(c_\tau, r_\tau, v_\tau) &= p(x_{1:F,\tau:T}|r_\tau, v_\tau) \\ &= p(x_{1:F,\tau}|c_\tau, r_\tau, v_\tau)\beta_{\tau|\tau+1}(c_\tau, r_\tau, v_\tau) \\ &= p(x_{1:F,\tau}|r_\tau, v_\tau)\beta_{\tau|\tau+1}(c_\tau, r_\tau, v_\tau) \qquad (4.20) \end{aligned}$$

where $\tau \in \{1, 2, ..., T-1\}$. Moreover, the posterior marginals can simply be obtained by multiplying the forward and backward messages:

$$\begin{aligned} p(c_\tau, r_\tau, v_\tau | x_{1:F,1:T}) &\propto p(x_{1:F,1:T}, c_\tau, r_\tau, v_\tau) \\ &= p(x_{1:F,1:\tau-1}, c_\tau, r_\tau, v_\tau) p(x_{1:F,\tau:T} | c_\tau, r_\tau, v_\tau, \cancel{x_{1:F,1:\tau-1}}) \\ &= \alpha_{\tau|\tau-1}(c_\tau, r_\tau, v_\tau) \beta_{\tau|\tau}(c_\tau, r_\tau, v_\tau). \end{aligned} \tag{4.21}$$

Due to the fact that r is discrete and v is continuous random variables, in the CPM, we have to store α and β messages as mixtures of Gamma distributions. In order to achieve ease of implementation, we can represent the Gamma mixture

$$p(v_\tau | r_\tau = i, \cdot) = \sum_{m=1}^{M} \exp(w_m) \mathcal{G}(v_\tau; a_m, b_m), \tag{4.22}$$

as $\{(a_1, b_1, w_1, i), (a_2, b_2, w_2, i), ..., (a_M, b_M, w_M, i)\}$. This will be simply $M \times 4$ array of parameters.

4.2.1. Forward Pass

To start the forward recursion, we define

$$\begin{aligned} \alpha_{0|0}(r_0, v_0) &= p(r_0, v_0) \\ &= p(r_0) p(v_0) \\ &= \sum_{i}^{I} \exp(l_i) \mathcal{G}(v_0; a_0, b_0) \end{aligned} \tag{4.23}$$

where, $l_i = \log p(r_0 = i)$. As we mentioned earlier, we represent this message with the array representation of the Gamma mixtures:

$$(a^k_{0|0}, b^k_{0|0}, c^k_{0|0}, d^k_{0|0}) = (a_0, b_0, l_k, k) \tag{4.24}$$

where $k = 1, 2, 3, ..., I$ denotes the index of the components in the Gamma mixture.

In the forward procedure, we have I Gamma potentials at time $\tau = 0$. Since we are dealing with the CPM, at each time frame, we would have two possibilities: there would be a change point or not. Hence, at $\tau = 1$, we would have I newly initialized Gamma potentials for the possibility of a change point and I Gamma potentials which we copy from the previous time frame, $\tau = 0$, in order to handle the case when a change point does not occur. Similarly, at $\tau = 2$, again we would have I newly initialized Gamma potentials to handle a change point and $2I$ Gamma potentials which we copy from $\tau = 1$. Note that we would have $(\tau + 1)I$ Gamma potentials at time frame τ. Figure 4.1 visualizes the procedure. Derivation of the prediction step at time τ is as follows:

$$\begin{aligned}
\alpha_{\tau|\tau-1}(c_\tau, r_\tau, v_\tau) &= \sum_{c_{\tau-1}} \sum_{r_{\tau-1}} \int dv_{\tau-1}\, p(c_\tau, r_\tau, v_\tau | r_{\tau-1}, v_{\tau-1}) \alpha_{\tau-1|\tau-1}(c_{\tau-1}, r_{\tau-1}, v_{\tau-1}) \\
&= \sum_{c_{\tau-1}} \sum_{r_{\tau-1}} \int dv_{\tau-1}\, p(v_\tau | c_\tau, r_\tau, v_{\tau-1}) p(r_\tau | c_\tau, r_{\tau-1}) p(c_\tau) \\
&\quad \alpha_{\tau-1|\tau-1}(c_{\tau-1}, r_{\tau-1}, v_{\tau-1}) \\
&= \Bigg(p(c_\tau = 1) \sum_{c_{\tau-1}} \sum_{r_{\tau-1}} \int dv_{\tau-1} \left(\mathcal{G}(v_\tau; a_v, b_v) p_1(r_\tau | r_{\tau-1}) \right) \\
&\quad + p(c_\tau = 0) \sum_{c_{\tau-1}} \sum_{r_{\tau-1}} \int dv_{\tau-1} \left(\delta(v_\tau - \theta(r_\tau) v_{\tau-1}) p_0(r_\tau | r_{\tau-1}) \right) \Bigg) \\
&\quad \alpha_{\tau-1|\tau-1}(c_{\tau-1}, r_{\tau-1}, v_{\tau-1}). \qquad (4.25)
\end{aligned}$$

The first I potentials that handle the change point case become

$$(a^k_{\tau|\tau-1}, b^k_{\tau|\tau-1}, c^k_{\tau|\tau-1}, d^k_{\tau|\tau-1}) = (a_v, b_v, c', k) \qquad (4.26)$$

for $k = 1, 2, ..., I$, where

$$c' = \log \left(\sum_{i=1}^{I} \sum_{j=1}^{I} [d^k_{\tau|\tau-1} = i] a^{(1)}_{ij} \sum_{m=1}^{\tau I} [d^m_{\tau-1|\tau-1} = j] \exp(c^m_{\tau-1|\tau-1}) \right) + \log w. \qquad (4.27)$$

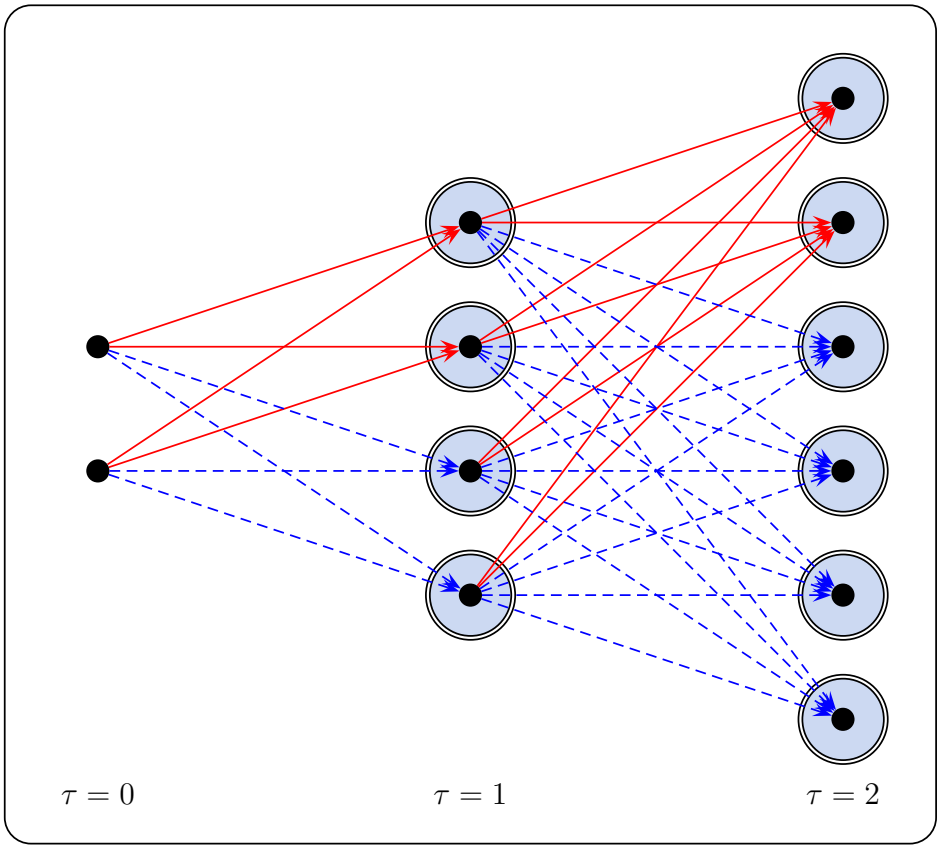


Figure 4.1. Visualization of the forward and the Viterbi algorithm for the CPM. Here, the number of templates, I is chosen to be 2. The small dots represent the Gamma potentials. For the forward procedure, the big circles represent the *sum* operator that sums the mixture coefficient of the Gamma potentials. For the Viterbi procedure, we replace the *sum* operator with the *max* operator which selects the Gamma potential that has the maximum mixture coefficient. The solid red arrows represent the case of the change point, and the dashed blue arrows represent the opposite case.

Here $a_{ij}^{(c)} = p_c(r_\tau = i | r_{\tau-1} = j)$.

We also have τI Gamma potentials which are inherited from the time frame $\tau - 1$:

$$(a^k_{\tau|\tau-1}, b^k_{\tau|\tau-1}, c^k_{\tau|\tau-1}, d^k_{\tau|\tau-1}) = (a^{k-I}_{\tau-1|\tau-1}, \frac{b^{k-I}_{\tau-1|\tau-1}}{\theta(d^{k-I}_{\tau-1|\tau-1})}, c', d^{k-I}_{\tau-1|\tau-1}) \tag{4.28}$$

for $k = I{+}1,\ I{+}2,...,\ (\tau+1)I$, where

$$c' = \log\left(\sum_{i=1}^{I}\sum_{j=1}^{I}[d^k_{\tau|\tau-1} = i]a^{(0)}_{ij}\sum_{m=1}^{\tau I}[d^m_{\tau-1|\tau-1} = j]\exp(c^m_{\tau-1|\tau-1})\right) + \log(1-w). \tag{4.29}$$

Once we compute the predictive distributions, we have to update the Gamma potentials as we acquire the observations:

$$\begin{aligned}
\alpha_{\tau|\tau}(c_\tau, r_\tau = i, v_\tau) &= p(x_{1:F,1:\tau}, r_\tau = i, v_\tau) \\
&= \alpha_{\tau|\tau-1}(c_\tau, r_\tau = i, v_\tau)p(x_{1:F,\tau}|c_\tau, r_\tau = i, v_\tau) \\
&= \sum_{m=1}^{(\tau+1)I}[d^m_{\tau|\tau-1} = i]\exp(c^m_{\tau|\tau-1})\mathcal{G}(v_\tau; a^m_{\tau|\tau-1}, b^m_{\tau|\tau-1}) \\
&\quad \prod_{\nu=1}^{F}\prod_{j=1}^{I}\mathcal{PO}(x_{\nu,\tau}; t_{\nu,j}v_\tau)^{[r_\tau=i]}.
\end{aligned} \tag{4.30}$$

Hence the update equation requires multiplication of Gamma and Poisson potentials. A nice property is that the product is also a Gamma potential, as derived in the Appendix A.3. The updated Gamma potentials are as follows:

$$(a^k_{\tau|\tau}, b^k_{\tau|\tau}, c^k_{\tau|\tau}, d^k_{\tau|\tau}) = (a', b', c', d') \tag{4.31}$$

for $k = 1, 2, ..., (\tau + 1)I$. Here

$$\begin{aligned} a' &= a^k_{\tau|\tau-1} + \sum_{\nu=1}^{F} x_{\nu,\tau} \\ b' &= b^k_{\tau|\tau-1} + \sum_{i=1}^{I} [d^k_{\tau|\tau-1} = i] \sum_{\nu=1}^{F} t_{\nu,i} \\ c' &= c^k_{\tau|\tau-1} + \sum_{i=1}^{I} [d^k_{\tau|\tau-1} = i] g(a^k_{\tau|\tau-1}, b^k_{\tau|\tau-1}, x, t) \\ d' &= d^k_{\tau|\tau-1}. \end{aligned} \tag{4.32}$$

4.2.2. Backward Pass

The backward pass is initialized as follows:

$$\beta_{T|T+1}(c_T, r_T, v_T) = 1 \tag{4.33}$$

$$(\hat{a}^k_{T|T+1}, \hat{b}^k_{T|T+1}, \hat{c}^k_{T|T+1}, \hat{d}^k_{T|T+1}) = (1, 0, 0, k), \tag{4.34}$$

for $k = 1, 2, ..., I$. Here the Gamma potential, $(1, 0, 0, k)$ is the improper Gamma distribution where

$$(a, b, c, k) \times (1, 0, 0, k) = (a, b, c, k), \tag{4.35}$$

for any a, b, and c.

Similar to the forward pass, we derive the backward recursion as follows:

$$\begin{aligned}
\beta_{\tau|\tau+1}(c_\tau, r_\tau, v_\tau) &= \sum_{c_{\tau+1}}\sum_{r_{\tau+1}} \int dv_{\tau+1}\, p(c_{\tau+1}, r_{\tau+1}, v_{\tau+1}|v_\tau, r_\tau)\beta_{\tau+1|\tau+1}(c_{\tau+1}, r_{\tau+1}, v_{\tau+1}) \\
&= \sum_{c_{\tau+1}}\sum_{r_{\tau+1}} \int dv_{\tau+1}\, p(v_{\tau+1}|r_{\tau+1}, c_{\tau+1}, v_\tau)p(r_{\tau+1}|c_{\tau+1}, r_\tau)p(c_{\tau+1}) \\
&\quad \beta_{\tau+1|\tau+1}(c_{\tau+1}, r_{\tau+1}, v_{\tau+1}) \\
&= \Bigg(p(c_\tau = 1) \sum_{c_{\tau+1}}\sum_{r_{\tau+1}} \int dv_{\tau+1}\, (\mathcal{G}(v_{\tau+1}; a_v, b_v)p_1(r_{\tau+1}|r_\tau)) \\
&\quad + p(c_\tau = 0) \sum_{c_{\tau+1}}\sum_{r_{\tau+1}} \int dv_{\tau+1}\, \big(\delta(v_{\tau+1} - \theta(r_{\tau+1})v_\tau)p_0(r_{\tau+1}|r_\tau)\big)\Bigg) \\
&\quad \beta_{\tau+1|\tau+1}(c_{\tau+1}, r_{\tau+1}, v_{\tau+1}). \qquad (4.36)
\end{aligned}$$

The first I potentials handle the change point case and the rest of them handle the opposite case:
for $k = 1, 2, ..., I$

$$(\hat{a}^k_{\tau|\tau+1}, \hat{b}^k_{\tau|\tau+1}, \hat{c}^k_{\tau|\tau+1}, \hat{d}^k_{\tau|\tau+1}) = (1, 0, c', k) \qquad (4.37)$$

for $k = I{+}1, I{+}2, ..., (T - \tau + 1)I$

$$(\hat{a}^k_{\tau|\tau+1}, \hat{b}^k_{\tau|\tau+1}, \hat{c}^k_{\tau|\tau+1}, \hat{d}^k_{\tau|\tau+1}) = (\hat{a}^{k-I}_{\tau+1|\tau+1}, \theta(r_{\tau+1})\hat{b}^{k-I}_{\tau+1|\tau+1}, c'', \hat{d}^{k-I}_{\tau+1|\tau+1}) \qquad (4.38)$$

where

$$\begin{aligned}
c' &= \log\left(\sum_{i=1}^{I}[\hat{d}^k_{\tau|\tau+1} = i]\sum_{j=1}^{I} a^{(1)}_{ji}\ \exp\kappa\Big(\mathcal{G}(v_\tau; \nu, B) \times \beta_{\tau+1|\tau+1}(c_{\tau+1}, r_{\tau+1} = j, v_{\tau+1})\Big)\right) \\
&\quad + \log w \qquad (4.39) \\
c'' &= \log\left(\sum_{i=1}^{I}[\hat{d}^k_{\tau|\tau+1} = i]\sum_{j=1}^{I} a^{(0)}_{ji}\ [\hat{d}^m_{\tau+1|\tau+1} = j]\sum_{m=1}^{(T-\tau)I} \exp \hat{c}^m_{\tau+1|\tau+1}\right) + \log(1 - w). \\
&\qquad (4.40)
\end{aligned}$$

The $\kappa(.)$ function returns the mixture coefficient of a Gamma potential:

$$\kappa(e^c \mathcal{G}(v; a, b)) = c, \tag{4.41}$$

and the $\times$ operator implies the product of two Gamma potentials which is presented in Appendix A.2 in more detail.

The backward recursions works very similar to the forward recursions, where we have I potentials at time T. At time $T-1$, we would have $2I$ Gamma potentials where the first I potentials handle the case of a change point and the remaning I potentials handle the opposite case which is the same case in the forward pass. Note that, in the backward pass we would have $(T-\tau+1)I$ Gamma potentials at time τ as opposed to the forward pass.

The update step at time τ is also similar to the forward pass:

$$\begin{aligned}
\beta_{\tau|\tau}(c_\tau, r_\tau, v_\tau) &= p(c_\tau, r_\tau, v_\tau, x_{1:F,\tau:T}) \\
&= p(x_{1:F,\tau}|r_\tau, v_\tau)\beta_{\tau|\tau+1}(c_\tau, r_\tau, v_\tau) \qquad (4.42) \\
\beta_{\tau|\tau}(c_\tau, r_\tau = i, v_\tau) &= \beta_{\tau|\tau+1}(c_\tau, r_\tau = i, v_\tau)p(x_{1:F,\tau}|v_\tau, r_\tau = i) \\
&= \sum_{m=1}^{(T-\tau+1)I} [\hat{d}^m_{\tau|\tau+1} = i] \exp(\hat{c}^m_{\tau|\tau+1})\mathcal{G}(v_\tau; \hat{a}^m_{\tau|\tau+1}, \hat{b}^m_{\tau|\tau+1}) \\
&\quad \prod_{\nu=1}^{F}\prod_{j=1}^{I} \mathcal{PO}(x_{\nu,\tau}; t_{\nu,j}v_\tau)^{[r_\tau=i]} \\
&= \sum_{m=1}^{(T-\tau+1)I} [\hat{d}^m_{\tau|\tau+1} = i] \exp(\hat{c}^m_{\tau|\tau+1})\mathcal{G}(v_\tau; \hat{a}^m_{\tau|\tau+1}, \hat{b}^m_{\tau|\tau+1}) \\
&\quad \prod_{\nu=1}^{F} \mathcal{PO}(x_{\nu,\tau}; t_{\nu,i}v_\tau) \qquad (4.43)
\end{aligned}$$

Since we are dealing with improper Gamma potential in the backward pass, the update step needs more attention in this case. The updated Gamma potentials are computed as follows:

for $k = 1, 2, ..., I$

$$(\hat{a}^k_{\tau|\tau}, \hat{b}^k_{\tau|\tau}, \hat{c}^k_{\tau|\tau}, \hat{d}^k_{\tau|\tau}) = (a', b', c', d') \tag{4.44}$$

for $k = I{+}1, I{+}2, ..., (T - \tau + 1)I$

$$(\hat{a}^k_{\tau|\tau}, \hat{b}^k_{\tau|\tau}, \hat{c}^k_{\tau|\tau}, \hat{d}^k_{\tau|\tau}) = (a'', b'', c'', d'') \tag{4.45}$$

where

$$\begin{aligned} a' &= \sum_{\nu=1}^{F} x_{\nu,\tau} \\ b' &= \sum_{i=1}^{I} [d^k_{\tau|\tau+1} = i] \sum_{\nu=1}^{F} t_{\nu,i} \\ c' &= \hat{c}^k_{\tau|\tau+1} + \sum_{i=1}^{I} [d^k_{\tau|\tau+1} = i] \hat{g}(x_{1:F,\tau}, t_{1:F,i}) \\ d' &= \hat{d}^k_{\tau|\tau+1}. \end{aligned} \tag{4.46}$$

and

$$\begin{aligned} a'' &= \hat{a}^k_{\tau|\tau+1} + \sum_{\nu=1}^{F} x_{\nu,\tau} \\ b'' &= \hat{b}^k_{\tau|\tau+1} + \sum_{i=1}^{I} [\hat{d}^k_{\tau|\tau+1} = i] \sum_{\nu=1}^{F} t_{\nu,i} \\ c'' &= \hat{c}^k_{\tau|\tau+1} + \sum_{i=1}^{I} [\hat{d}^k_{\tau|\tau+1} = i] g(\hat{a}^k_{\tau|\tau+1}, \hat{b}^k_{\tau|\tau+1}, x_{1:F,\tau}, t_{1:F,i}) \\ d'' &= \hat{d}^k_{\tau|\tau+1}. \end{aligned} \tag{4.47}$$

The details of the derivations are presented in Appendix A.3 and A.4.

4.2.3. Smoothing

Once we compute the forward and backward potentials, the smoothed posterior can be obtained by computing the product of the potentials as follows:

$$\begin{aligned}
p(c_\tau, r_\tau, v_\tau | x_{1:F,1:T}) &\propto p(x_{1:F,1:T}, c_\tau, r_\tau, v_\tau) \\
&= p(x_{1:F,1:\tau-1}, c_\tau, r_\tau, v_\tau) p(x_{1:F,\tau:T} | \cancel{c_\tau}, r_\tau, v_\tau, \cancel{x_{1:F,1:\tau-1}}) \\
&= p(x_{1:F,1:\tau}, v_\tau, r_\tau) p(x_{1:F,\tau+1:T} | \cancel{c_\tau}, r_\tau, v_\tau, \cancel{x_{1:F,1:\tau}}) \\
&= \alpha_{\tau|\tau-1}(c_\tau, r_\tau, v_\tau) \beta_{\tau|\tau}(c_\tau, r_\tau, v_\tau) \\
&= \alpha_{\tau|\tau}(c_\tau, r_\tau, v_\tau) \beta_{\tau|\tau+1}(c_\tau, r_\tau, v_\tau), \qquad (4.48) \\
p(c_\tau, r_\tau = i, v_\tau | x_{1:F,1:T}) &\propto \alpha_{\tau|\tau-1}(c_\tau, r_\tau = i, v_\tau) \beta_{\tau|\tau}(c_\tau, r_\tau = i, v_\tau) \\
&= \alpha_{\tau|\tau}(c_\tau, r_\tau = i, v_\tau) \beta_{\tau|\tau+1}(c_\tau, r_\tau = i, v_\tau). \qquad (4.49)
\end{aligned}$$

4.2.4. Marginal Viterbi Path

The marginal Viterbi path is defined as:

$$\begin{aligned}
(c^*_{1:T}, r^*_{1:T}) &= \underset{c_{1:T}, r_{1:T}}{\operatorname{argmax}} \int_{v_{1:T}} p(x_{1:F,1:T}, c_{1:T}, r_{1:T}, v_{1:T}) \\
&= \underset{c_{1:T}, r_{1:T}}{\operatorname{argmax}} \int_{v_{1:T}} \gamma(c_{1:T}, r_{1:T}, v_{1:T}), \qquad (4.50)
\end{aligned}$$

where $\gamma(c_\tau, r_\tau, v_\tau)$ corresponds to the smoothed Gamma potentials at time τ. However, as opposed to Equation (4.12), replacing the summations over r_τ and c_τ by maximization can be problematic since maximization and integration do not commute. We sum and integrate over the hidden variables v_τ first. In other words we compute the mixture coefficients of the Gamma potentials, and then select the maximum of them. We call this path as "marginal", since in order to achieve the exact Viterbi path, we should have also replaced the integration over v_τ by maximization in Equation (4.50). Fortunately, for this model, we are able to compute the exact marginal distribution of r_τ

and c_τ, $p(c_{1:T}, r_{1:T}|x_{1:F,1:T})$, and the exact marginal Viterbi path. This is not the case for many probabilistic models (Cemgil *et al.*, 2006).

For example, suppose we have six Gamma potentials:

$$\gamma(c, r, v) = \{\gamma_0^1, \gamma_0^2, \gamma_0^3, \gamma_1^1, \gamma_1^2, \gamma_1^3\} \tag{4.51}$$

where $\gamma_i^j = \gamma(c = i, r = j, v)$. Then we can compute the MAP (maximum a-posteriori) configuration:

$$\begin{aligned}(c^*, r^*) &= \underset{c,r}{\operatorname{argmax}} \int_v \gamma(c, r, v) \\ &= \underset{c,r}{\operatorname{argmax}} \{k_0^1, k_0^2, \ldots, k_1^3\}.\end{aligned} \tag{4.52}$$

Here k_i^j are the normalization constants of the Gamma potentials, where $k_i^j = \kappa(\gamma_i^j)$ (see Equation (4.41)). Figure 4.1 visualizes the procedure.

4.2.5. Approximations

One disadvantage of this model is that the need for the computational power increases as τ increases and exact inference becomes impractical after a couple of steps. In order to eliminate this problem we developed a pruning technique for the CPM as an approximate inference scheme. In the standard pruning algorithms, at time τ, we would sort the Gamma potentials with respect to their mixture coefficients $c_{\tau|\tau}^k$, keep the N best potentials, and discard the rest of them. However, with this scheme, we may unwillingly discard the first, immature potentials in the mixture since they have been recently inserted to the mixture.

In this study we propose a different pruning scheme for the CPM. As opposed to the standard pruning methods, we always keep the first N_{keep} Gamma potentials without taking into account their mixture coefficients, where $0 \leq N_{keep} \ll N$. Then we apply the standard pruning algorithm to the rest of the potentials, i.e. we select the

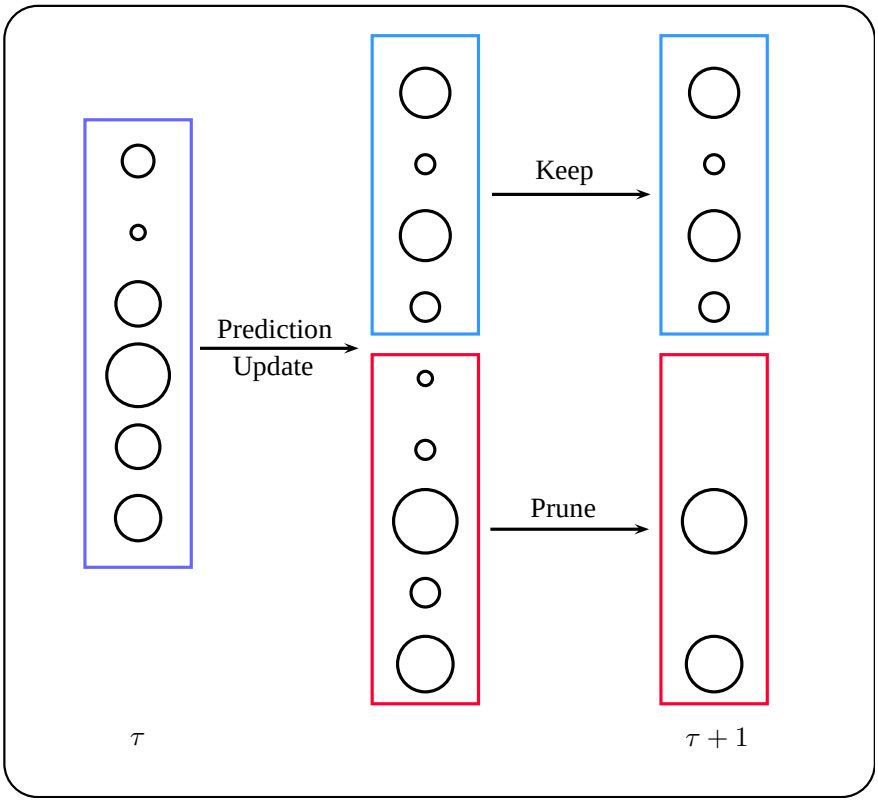


Figure 4.2. Illustration of the pruning schema of the CPM. The circles represent the Gamma potentials where the radius of a circle is proportional to the mixture coefficient of the potential. In this example $N = 6$ and $N_{keep} = 4$.

$(N - N_{keep})$ best Gamma potentials. An illustration of the pruning schema is presented in Figure 4.2.

5. TRAINING AND PARAMETER LEARNING

Since we have constructed our inference algorithms with the assumption of the templates $t_{\nu,i}$ to be known, we have to learn the spectral templates at the beginning. In this study we utilized the EM algorithm for this purpose. This algorithm iteratively maximizes the log-likelihood.

Consider we have the likelihood $p(y|\theta)$ and we want to find the θ which maximizes the likelihood. More formally, we can write this statement as:

$$\underset{\theta}{\operatorname{argmax}}\, p(y|\theta) = \underset{\theta}{\operatorname{argmax}}\, \log p(y|\theta) = \underset{\theta}{\operatorname{argmax}}\, \mathcal{L}(\theta), \tag{5.1}$$

where $\mathcal{L}(\theta)$ is the *log-likelihood* and can be written as:

$$\begin{aligned} \mathcal{L}(\theta) &= \log \sum_x p(y,x|\theta)\frac{q(x)}{q(x)} \\ &= \log \left\langle \frac{p(y,x|\theta)}{q(x)} \right\rangle_{q(x)}, \end{aligned} \tag{5.2}$$

where $q(x)$ is an instrumental distribution and $\langle f(x)\rangle_{p(x)}$ is the expectation of the function $f(x)$ with respect to $p(x)$:

$$\langle f(x)\rangle_{p(x)} = \int p(x)f(x)dx. \tag{5.3}$$

By using Jensen's inequality we get

$$\log \left\langle \frac{p(y,x|\theta)}{q(x)} \right\rangle_{q(x)} \geq \left\langle \log \frac{p(y,x|\theta)}{q(x)} \right\rangle_{q(x)} = \mathcal{B}[q]. \tag{5.4}$$

Here $\mathcal{B}[q]$ is a lower bound on the log-likelihood and we want to find the $q(x)$ which maximizes this lower bound. By utilizing the constraint, $\sum_x q(x) = 1$ via Lagrange

multipliers, we obtain the following objective:

$$\Lambda = \sum_x q(x) \log p(x, y|\theta) - \sum_x q(x)q(x) + \lambda(1 - \sum_x q(x)). \tag{5.5}$$

By maximizing the objective with respect to $q(x)$ we get

$$\frac{\partial \Lambda}{\partial q} = \log p(y, x|\theta) - (\log q(x) + 1) - \lambda = 0 \tag{5.6}$$

$$\log q(x) = \log p(y, x|\theta) - 1 - \lambda \tag{5.7}$$

$$q(x) = p(y, x|\theta) \exp(-1 - \lambda) \tag{5.8}$$

$$1 = p(y|\theta) \exp(-1 - \lambda). \tag{5.9}$$

By combining Equations (5.8) and (5.9), we can derive the optimum $q(x)$:

$$\begin{aligned} q^*(x) &= \frac{p(x, y|\theta)}{p(y|\theta)} \\ &= p(x|y, \theta) \end{aligned} \tag{5.10}$$

which is the posterior distribution. We can rewrite the lower bound in Equation (5.4) by using the result that is obtained from Equation (5.10):

$$\begin{aligned} \mathcal{B}^*[q] &= \langle \log p(x, y|\theta) \rangle_{p(x|y,\theta)} - \left\langle \log p(x|y, \theta^{old}) \right\rangle_{p(x|y,\theta^{old})} \\ &= \langle \log p(x, y|\theta) \rangle_{p(x|y,\theta)} + \mathcal{H}[q], \end{aligned} \tag{5.11}$$

where $\mathcal{H}[q]$ is the entropy term which can be ignored during maximization. Hence, for our probabilistic models, the log-likelihood can be maximized iteratively as follows:

E-step :

$$q(c_{1:T}, r_{1:T}, v_{1:T})^{(n)} = p(c_{1:T}, r_{1:T}, v_{1:T}|x_{1:F,1:T}, t_{1:F,1:I}^{(n-1)}) \tag{5.12}$$

M-step :

$$t_{1:F,1:I}^{(n)} = \underset{t_{1:F,1:I}}{\operatorname{argmax}} \left\langle p(c_{1:T}, r_{1:T}, v_{1:T}, x_{1:F,1:T}|t_{1:F,1:I}^{(n)}) \right\rangle_{q(c_{1:T}, r_{1:T}, v_{1:T})^{(n)}}. \tag{5.13}$$

In the E-step, we compute the posterior distributions of c_τ, r_τ and v_τ. These quantities can be computed via the methods which we described in Section 3.1 and 3.2 for the HMM and the CPM respectively. In the M-step, which is a fixed point equation, we want to find the $t_{\nu,i}$ that maximize the likelihood. The M-step for both models is computed as follows: We first write the posterior

$$\begin{aligned} p(c_{1:T}, r_{1:T}, v_{1:T}, x_{1:F,1:T}|t^{(n-1)}_{1:F,1:I}) &= p(x_{1:F,1:T}|\cancel{c_{1:T}}, r_{1:T}, v_{1:T}, t^{(n-1)}_{1:F,1:I}) p(v_{1:T}, r_{1:T}|\cancel{t^{(n-1)}_{1:F,1:I}}) \\ &= \left(\prod_{\nu=1}^{F} \prod_{\tau=1}^{T} \prod_{i=1}^{I} \mathcal{PO}(x_{\nu,\tau}; t_{\nu,i} v_\tau)^{[r_\tau = i]} \right) p(c_{1:T}, r_{1:T}, v_{1:T}). \end{aligned} \tag{5.14}$$

Then we define our objective by eliminating the terms that are not dependent on $t_{\nu,i}$:

$$\begin{aligned} Q &= \sum_{\nu=1}^{F} \sum_{\tau=1}^{T} \sum_{i=1}^{I} [r_\tau = i](x_{\nu,\tau} \log t_{\nu,i} v_\tau - t_{\nu,i} v_\tau) + \cancel{\log p(c_{1:T}, r_{1:T}, v_{1:T})} \\ &= \sum_{\nu=1}^{F} \sum_{\tau=1}^{T} \sum_{i=1}^{I} [r_\tau = i] x_{\nu,\tau} \log t_{\nu,i} + \cancel{[r_\tau = i] x_{\nu,\tau} \log v_\tau} - t_{\nu,i} [r_\tau = i] v_\tau \\ &= \sum_{\nu=1}^{F} \sum_{\tau=1}^{T} \sum_{i=1}^{I} [r_\tau = i] x_{\nu,\tau} \log t_{\nu,i} - t_{\nu,i} [r_\tau = i] v_\tau. \end{aligned} \tag{5.15}$$

By maximizing the objective Q and substituting $\langle [r_\tau = i] \rangle^{(n)}$ and $\langle [r_\tau = i] v_\tau \rangle^{(n)}$, we obtain the following equations:

$$\frac{\partial Q}{\partial t_{\nu,i}} = \frac{\sum_{\tau=1}^{T} \langle [r_\tau = i] \rangle^{(n)} x_{\nu,\tau}}{t_{\nu,i}} - \sum_{\tau=1}^{T} \langle [r_\tau = i] v_\tau \rangle^{(n)} \tag{5.16}$$

$$t^{(n)}_{\nu,i} = \frac{\sum_{\tau=1}^{T} \langle [r_\tau = i] \rangle^{(n)} x_{\nu,\tau}}{\sum_{\tau=1}^{T} \langle [r_\tau = i] v_\tau \rangle^{(n)}}. \tag{5.17}$$

Intuitively, we can interpret this result as the weighted average of the normalized audio spectra with respect to v_τ.

6. EVALUATION AND RESULTS

In order to evaluate the performance of the probabilistic models on pitch tracking, we have conducted several experiments. As mentioned earlier, in this study we focus on the monophonic pitch tracking of low-pitched instruments.

In our experiments we used the electric bass guitar and tuba recordings of the RWC Musical Instrument Sound Database. We first trained the templates offline in MATLAB environment, and then we tested our models by utilizing the previously learned templates in C++ environment.

At the training step, we ran the EM algorithm for each note where we used short isolated recordings. On the whole, we use 28 recordings for bass guitar (from E2 to G4) and 27 recordings for tuba (from F2 to G4). The HMM's training phase lasts approximately 30 seconds and the CPM's lasts approximately 2 minutes.

At the testing step, we rendered monophonic MIDI files to audio by using the samples from RWC recordings. The total duration of the test files are approximately 5 minutes. At the evaluation step, we compared our estimates with the ground truth which is obtained from the MIDI file. In both our models we used 46 ms. long frames at 44.1 kHz sampling rate. Figure 6.1 shows excerpts from the test files.

In our point of view, the main trade-off of these pitch tracking models is between the latency and the accuracy. We can increase the accuracy by accumulating the data, in other words increasing the latency. However after some point the pitch tracking system would be useless due to the high latency. Hence we tried to find the optimum latency and accuracy by adjusting the "lag" parameter of the fixed-lag viterbi path which is defined in Equation (3.4).

As evaluation metrics, we used the recall rate, the precision rate, the speed factor and the note onset latency. The recall rate, the precision rate and the speed factor is

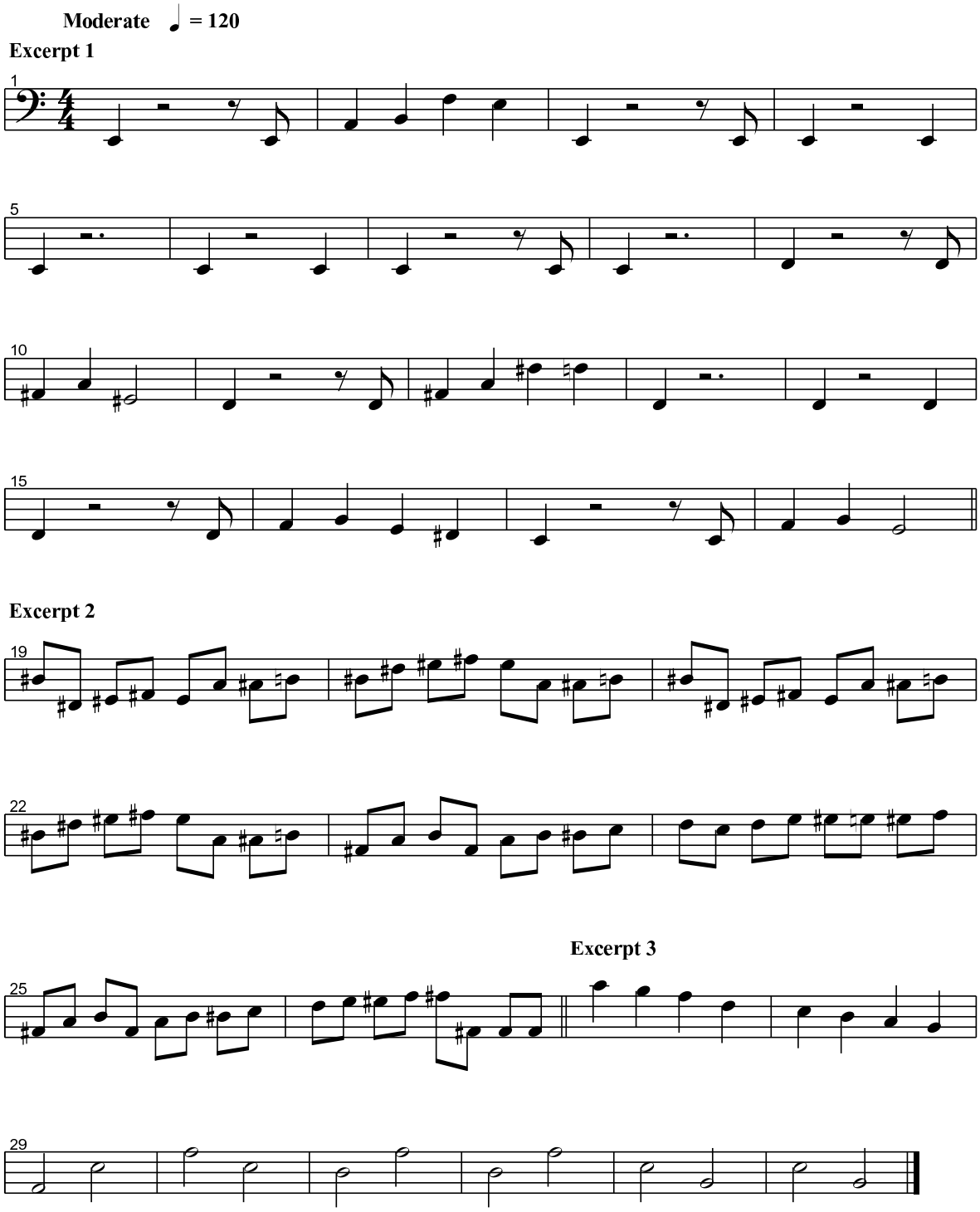


Figure 6.1. Excerpts of the test files.

defined in Table 6.1 and we define the note onset latency as the time difference between the pitch tracker's estimate and the ground truth, without considering the label of the estimate.

Table 6.1. Definition of the evaluation metrics. Note that the *speed factor* is a cpu-dependent metric which would be lower in faster computers.

recall	$\frac{\text{num. of correct notes}}{\text{num. of true notes}}$
precision	$\frac{\text{num. of correct notes}}{\text{num. of transcribed notes}}$
speed factor	$\frac{\text{running time of the method}}{\text{duration of the test file}}$

6.1. Performance of the HMM

As we mentioned in Subsection 3.1.1, we used one single state for silence and three states for each note. Figure 6.2 presents the transition matrix of r_τ for 22 states (silence and 7 notes). The indexing structure of the matrix is given in Table 6.2. The overall performance of the HMM is shown in Figure 6.3.

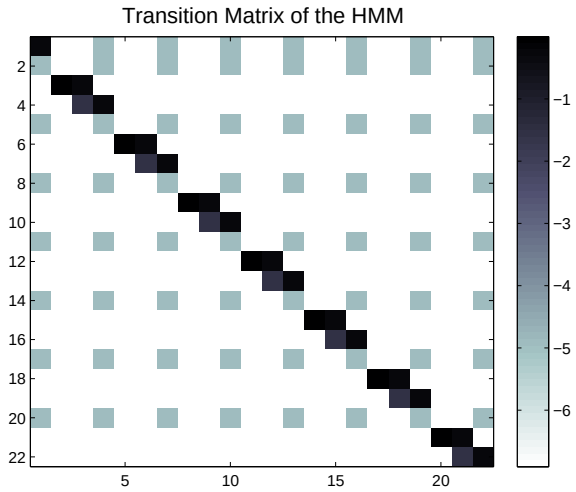


Figure 6.2. Logarithm of the transition matrix of the HMM.

Table 6.2. The indexing structure in the state transition matrix.

State index	Equivalent
1	Silence
2	Note 1 - attack part
3	Note 1 - sustain part
4	Note 1 - release part
5	Note 2 - attack part
6	Note 2 - sustain part
7	Note 2 - release part
8	Note 3 - attack part
⋮	⋮
22	Note 7 - release part
⋮	⋮

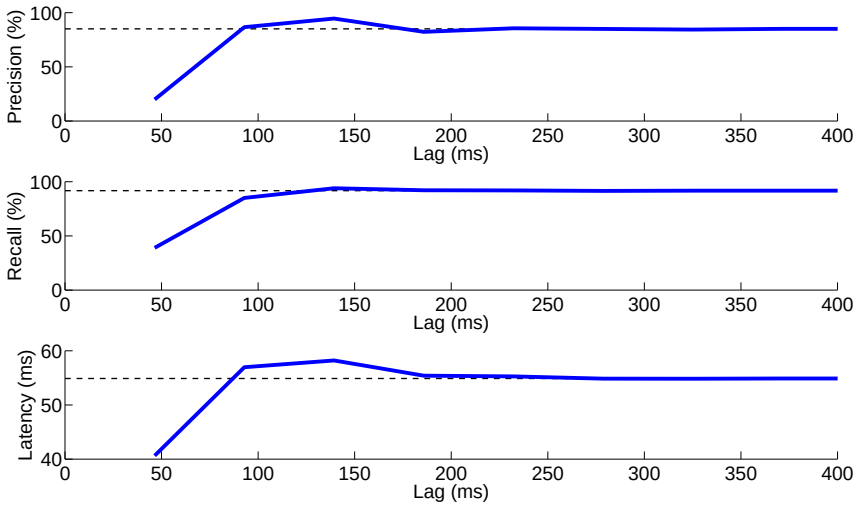
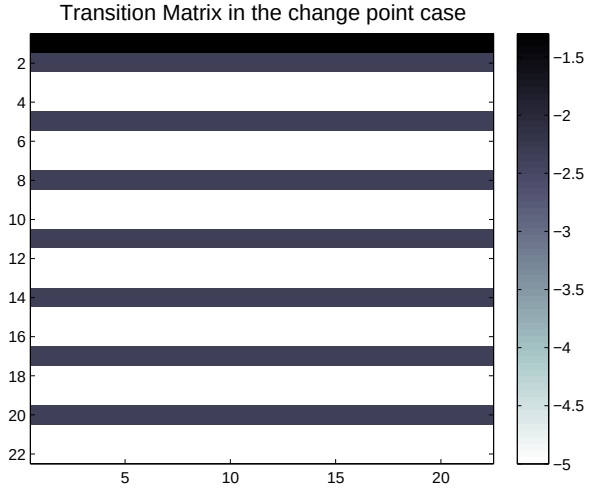


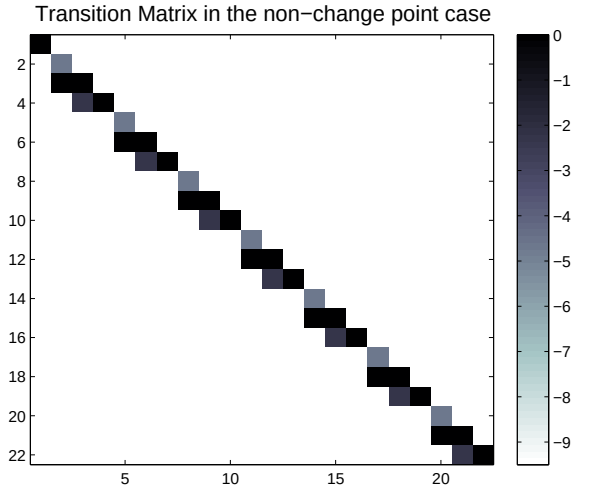
Figure 6.3. The overall performance of the HMM on low-pitched audio. The dashed lines represent the offline processing results. The total latency of the system is the sum of the lag and the latency at the note onsets.

6.2. Performance of the CPM

In the CPM, we have two different state transition matrices of r_τ. The first matrix becomes active in the case of a change point, and the second matrix becomes active in the opposite case. Figure 6.4 visualizes the transition matrices that we used in our tests. The indexing structure is given in Table 6.2. The overall performance of the CPM is shown in Figure 6.5.



(a) Logarithm of the transition matrix of the CPM for the change point case



(b) Logarithm of the transition matrix of the CPM for the non-change point case

Figure 6.4. Logarithm of the transition matrices of the CPM.

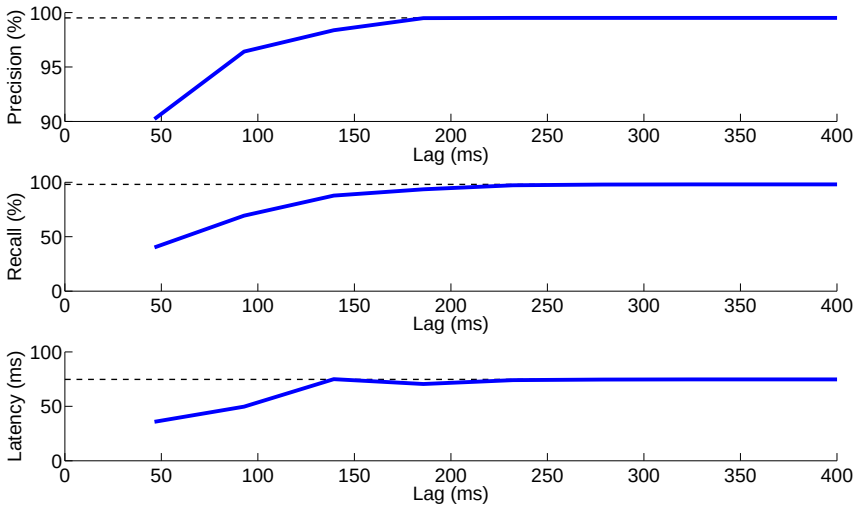


Figure 6.5. The overall performance of the CPM on low-pitched audio. The dashed lines represent the offline processing results. The total latency of the system is the sum of the lag and the latency at the note onsets.

6.3. Comparison with the YIN algorithm

We also compared the performance of our models with the well-known YIN algorithm (de Cheveigné and Kawahara, 2002). Despite the fact that YIN is a general purpose method, we compared our results with the YIN's, since YIN is accepted as a standard method for pitch tracking. We used the *aubio* implementation and tuned the onset threshold parameter. The results are shown in Table 6.3.

Table 6.3. The comparison of our models with the YIN algorithm. The CPM performs better than the others. Moreover, the HMM would also be advantageous due to its cheaper computational needs.

	Recall (%)	Precision (%)	Onset Latency (ms)	Speed Factor
YIN	43.43	9.40	58.74	1.33
HMM	91.72	85.02	54.89	0.02
CPM	98.06	99.50	74.74	0.05

7. DISCUSSION, CONCLUSIONS, AND FUTURE WORK

In this study we presented and compared two probabilistic models for online pitch tracking. The motivation was to develop an accurate monophonic pitch tracking method which would be quite useful for the musicians who play low-pitched instruments (like the bass guitar or the tuba) since the melodies that are played with these instruments are mostly monophonic due to the nature of these instruments. Hence, our main focus was on monophonic pitch tracking of low-pitched instruments and the trade off between latency and accuracy of the proposed pitch tracking models.

Apart from the previous works that aimed to develop instrument-independent pitch tracking systems, our approach is based on modeling of a specific musical instrument's spectral structure. Our models can be fine-tuned for any instrument with a quick training procedure.

In our models, it is presumed that each note has a certain characteristic spectral shape which we call the *spectral template*. The generative models were constructed in such a way that each time slice of the audio spectra is generated from one of these spectral templates multiplied by a volume factor. From this point of view, we treated the pitch tracking problem as a template matching problem where the aim is to infer which template is active and what is the volume as we observe the audio data.

In the HMM, we assumed that the pitch labels have a certain temporal structure in such a way that the current pitch label depended on the previous pitch label. In order to preserve the structure of the HMM, we had to let the volume variables to be independent in time, which is not the natural case in terms of musical audio. However, in this case, the inference scheme turned out to be quite simple, straightforward, and hereby fast.

In addition to the temporal structure of the pitch labels that we introduced at the HMM, in the CPM, we introduced a temporal structure also for the volume variables.

In this way, the CPM was able to model of the damping structure of an instrument. As a trade off, the inference scheme of the CPM was much more complex than the HMM. Besides, after some degree, exact inference became impractical. Hence, an approximate inference scheme was developed for this model.

The main focus on this work was the trade off between latency and accuracy of the pitch tracking system. We conducted several experiments on an implementation which was developed in C++, in order to find the optimum accuracy and latency. We evaluated the performance of our models by computing the most-likely paths that were obtained via filtering or fixed-lag smoothing distributions. The evaluation was held on monophonic bass guitar and tuba recordings with respect to four evaluation metrics. We also compared the results with the YIN algorithm and obtained better results.

One limitation of the CPM is that it has the same damping coefficient (θ) for all frequency components in the spectrum. This assumption is limiting since each frequency component of a note evolves differently over time. As a natural next step of our work is to construct probabilistic models that have frequency dependent damping coefficients.

One advantage of probabilistic modeling is that it is possible to combine the proposed models with different kinds of probabilistic models. This enables deeper music analysis schemes. For instance, these models can be combined with tempo tracking models for joint pitch-tempo tracking, to name a few: (Whiteley *et al.*, 2006; Whiteley *et al.*, 2007). In such joint models, the melody information would also be used while tracking the tempo and vice versa. This idea is not yet studied in the literature extensively and may end up with interesting results.

Despite testing our models on monophonic data, the models are also extensible to more complicated scenarios such as polyphony. This can be done by using factorial models (Cemgil, 2006) or using hierarchical NMF models where in this case r_τ and v_τ would be vectors instead of scalars. This kind of extensions require more com-

plex inference schemes, but fortunately there exists powerful state-of-the-art inference methods.

On the other hand, this template matching framework can also be used for various types of applications since we do not make any application specific assumptions. One other next step of this work will be testing this framework on real-time detection of acoustic events based on the work (Jylhä and Erkut, 2008). Thanks to the flexibility of the models, for acoustic event detection, we only need to replace the spectral templates of the notes with the spectral templates of acoustic events.

APPENDIX A: OPERATIONS ON GAMMA POTENTIALS

A.1. Derivation of the Compound Poisson Observation Model

In the HMM, we can analytically integrate out the volume variables. This lets us to have the *Compound Poisson* observation model which only depends on the indicator variables, r_τ. The derivation is as follows:

$$\begin{aligned}
p(x_{1:F,1:\tau}|r_\tau) &= \int dv_\tau \; p(x_{1:F,1:\tau}, v_\tau|r_\tau) \\
&= \int dv_\tau \; p(x_{1:F,1:\tau}|v_\tau, r_\tau)p(v_\tau|\not{r_\tau}) \\
&= \int dv_\tau \; \prod_{\nu=1}^{F}\prod_{i=1}^{I} \mathcal{PO}(x_{\nu,\tau}; v_\tau t_{\nu,i})^{[r_\tau=i]}\mathcal{G}(v_\tau; a, b) \\
&= \int dv_\tau \; \exp\left(\sum_{\nu=1}^{F}\sum_{i=1}^{I}[r_\tau = i]\log\mathcal{PO}(x_{\nu,\tau}; v_\tau t_{\nu,i}) + \log\mathcal{G}(v_\tau; a, b)\right)
\end{aligned} \tag{A.1}$$

$$\begin{aligned}
p(x_{1:F,1:\tau}|r_\tau = i) &= \int dv_\tau \; \exp\left(\sum_{\nu=1}^{F}\log\mathcal{PO}(x_{\nu,\tau}; v_\tau t_{\nu,i}) + \log\mathcal{G}(v_\tau; a, b)\right) \\
&= \int dv_\tau \; \exp\left(\left(\sum_{\nu=1}^{F} x_{\nu,\tau}\log v_\tau t_{\nu,i} - v_\tau t_{\nu,i} - \log\Gamma(x_{\nu,\tau}+1)\right)\right. \\
&\quad \left.+\left((a-1)\log v_\tau - bv_\tau - \log\Gamma(a) + a\log(b)\right)\right) \\
&= \int dv_\tau \; \exp\left((\sum_{\nu=1}^{F} x_{\nu,\tau} + a - 1)\log v_\tau - (\sum_{\nu=1}^{F} t_{\nu,i} + b)v_\tau\right. \\
&\quad - \log\Gamma(\sum_{\nu=1}^{F} x_{\nu,\tau} + a) + (\sum_{\nu=1}^{F} x_{\nu,\tau} + a)\log(\sum_{\nu=1}^{F} t_{\nu,i} + b) \\
&\quad + \log\Gamma(\sum_{\nu=1}^{F} x_{\nu,\tau} + a) - (\sum_{\nu=1}^{F} x_{\nu,\tau} + a)\log(\sum_{\nu=1}^{F} t_{\nu,i} + b) \\
&\quad \left.- \sum_{\nu=1}^{F}\log\Gamma(x_{\nu,\tau}+1) + \sum_{\nu=1}^{F} x_{\nu,\tau}\log t_{\nu,i} - \log\Gamma(a) + a\log(b)\right)
\end{aligned}$$

$$
\begin{aligned}
&= \exp\Bigg(\log\Gamma(\sum_{\nu=1}^{F} x_{\nu,\tau} + a) - (\sum_{\nu=1}^{F} x_{\nu,\tau} + a)\log(\sum_{\nu=1}^{F} t_{\nu,i} + b) \\
&\quad - \sum_{\nu=1}^{F} \log\Gamma(x_{\nu,\tau}+1) + \sum_{\nu=1}^{F} x_{\nu,\tau}\log t_{\nu,i} - \log\Gamma(a) + a\log(b) \Bigg) \\
&\quad \cancel{\int dv_\tau \, \mathcal{G}(v_\tau; \sum_{\nu=1}^{F} x_{\nu,\tau} + a, \sum_{\nu=1}^{F} t_{\nu,i} + b)} \\
&= \frac{\Gamma(\sum_{\nu=1}^{F} x_{\nu,\tau} + a)}{\Gamma(a)\prod_{\nu=1}^{F}\Gamma(x_{\nu,\tau}+1)} \frac{b^a \prod_{\nu=1}^{F} t_{\nu,i}^{x_{\nu,\tau}}}{(\sum_{\nu=1}^{F} t_{\nu,i} + b)^{\sum_{\nu=1}^{F} x_{\nu,\tau} + a}}
\end{aligned} \tag{A.2}
$$

A.2. Products of two Gamma potentials

While making inference on the CPM, we need to compute the product of Gamma potentials in the backward procedure. A nice property is that the product is also a Gamma potential which is derived as follows:

$$
\begin{aligned}
f_1(x) &= e^{c_1}\mathcal{G}(x; a_1, b_1) \\
f_2(x) &= e^{c_2}\mathcal{G}(x; a_2, b_2)
\end{aligned} \tag{A.3}
$$

$$
\begin{aligned}
\log f_1(x) f_2(x) =& c_1 + (a_1 - 1)\log x - b_1 x - \log\Gamma(a_1) + a_1\log(b_1) + c_2 \\
&+ (a_2 - 1)\log x - b_2 x - \log\Gamma(a_2) + a_2\log(b_2) \\
=& (a_1 + a_2 - 1 - 1)\log x - (b_1 + b_2)x + c_1 + c_2 - \log\Gamma(a_1) \\
&- \log\Gamma(a_2) \\
=& + a_1\log(b_1) + a_2\log(b_2) \log\mathcal{G}(x; a_1 + a_2 - 1, b_1 + b_2) + c_1 + c_2 \\
&+ \log\Gamma(a_1 + a_2 - 1) - (a_1 + a_2 - 1)\log(b_1 + b_2) - \log\Gamma(a_1) \\
&- \log\Gamma(a_2) + a_1\log(b_1) + a_2\log(b_2)
\end{aligned}
$$

$$
\begin{aligned}
&= \log \mathcal{G}(x; a_1 + a_2 - 1, b_1 + b_2) + c_1 + c_2 + \log \Gamma(a_1 + a_2 - 1) \\
&\quad - \log \Gamma(a_1) - \log \Gamma(a_2) + \log(b_1 + b_2) + a_1 \log(b_1/(b_1 + b_2)) \\
&\quad + a_2 \log(b_2/(b_1 + b_2)) \\
&= \log\Big(e^{c'} \mathcal{G}(x; a', b')\Big)
\end{aligned} \tag{A.4}
$$

where

$$
\begin{aligned}
a' =& a_1 + a_2 - 1 \\
b' =& b_1 + b_2 \\
c' =& c_1 + c_2 + \log \Gamma(a_1 + a_2 - 1) - \log \Gamma(a_1) - \log \Gamma(a_2) + \log(b_1 + b_2) \\
&+ a_1 \log(b_1/(b_1 + b_2)) + a_2 \log(b_2/(b_1 + b_2)).
\end{aligned} \tag{A.5}
$$

A.3. Update Step of a Single Gamma Potential

While making inference on the CPM, we need to update the Gamma potentials in the forward and backward procedures as we observe the audio data. The update step of a single Gamma potential is derived as follows:

$$
\begin{aligned}
&= \log \left(\exp(c) \mathcal{G}(v_\tau; a, b) \prod_{\nu=1}^{F} \mathcal{PO}(x_{\nu,\tau}; t_{\nu,i} v_\tau) \right) \\
&= c + \log \mathcal{G}(v_\tau; a, b) + \sum_{\nu=1}^{F} \log \mathcal{PO}(x_{\nu,\tau}; t_{\nu,i} v_\tau) \\
&= c + (a - 1) \log v_\tau - b v_\tau - \log \Gamma(a) + a \log(b) \\
&\quad + \sum_{\nu=1}^{F} (x_{\nu,\tau} \log t_{\nu,i} v_\tau - t_{\nu,i} v_\tau - \log \Gamma(x_{\nu,\tau} + 1)) \\
&= (a + X_\tau - 1) \log v_\tau - (b + T_i) v_\tau + c - \log \Gamma(a) + a \log(b) + \sum_{\nu=1}^{F} x_{\nu,\tau} \log t_{\nu,i} \\
&\quad - \sum_{\nu=1}^{F} \log \Gamma(x_{\nu,\tau} + 1)
\end{aligned}
$$

$$
\begin{aligned}
=&(a + X_\tau - 1)\log v_\tau - (b + T_i)v_\tau - \log\Gamma(a + X_\tau) + (a + X_\tau)\log(b + T_i) \\
&+ c + g(a, b, x, t) \\
=&c' + \log\mathcal{G}(v_\tau; a', b'),
\end{aligned} \tag{A.6}
$$

where

$$
\begin{aligned}
X_\tau =& \sum_{\nu=1}^{F} x_{\nu,\tau} \\
T_i =& \sum_{\nu=1}^{F} t_{\nu,i} \\
a' =& a + X_\tau \\
b' =& b + T_i \\
c' =& c + g(a, b, x, t)
\end{aligned} \tag{A.7}
$$

and

$$
\begin{aligned}
g(.) =& \log\Gamma(a + X_\tau) - (a + X_\tau)\log(b + T_i) - \log\Gamma(a) + a\log(b) \\
&+ \sum_{\nu=1}^{F} x_{\nu,\tau}\log t_{\nu,i} - \sum_{\nu=1}^{F}\log\Gamma(x_{\nu,\tau} + 1).
\end{aligned} \tag{A.8}
$$

A.4. Update Step of An Improper Gamma Potential

While making inference on the CPM, we need to update improper Gamma potentials in the backward procedure as we observe the audio data. The update step of a single improper Gamma potential is derived as follows:

$$
\begin{aligned}
=& \log\left(\exp(c)\cancel{\mathcal{G}(v_\tau; 1, 0)}\prod_{\nu=1}^{F}\mathcal{PO}(x_{\nu,\tau}; t_{\nu,i}v_\tau)\right) \\
=& c + \sum_{\nu=1}^{F}\log\mathcal{PO}(x_{\nu,\tau}; t_{\nu,i}v_\tau)
\end{aligned}
$$

$$
\begin{aligned}
=&c+\sum_{\nu=1}^{F} x_{\nu,\tau}\log t_{\nu,i}v_\tau - t_{\nu,i}v_\tau - \log\Gamma(x_{\nu,\tau}+1)\\
=&c+\sum_{\nu=1}^{F} x_{\nu,\tau}\log t_{\nu,i} + x_{\nu,\tau}\log v_\tau - t_{\nu,i}v_\tau - \log\Gamma(x_{\nu,\tau}+1)\\
=&(\sum_{\nu=1}^{F} x_{\nu,\tau}-1)\log v_\tau - (\sum_{\nu=1}^{F} t_{\nu,i})v_\tau + c + \sum_{\nu=1}^{F} x_{\nu,\tau}\log t_{\nu,i} - \log\Gamma(x_{\nu,\tau}+1)\\
=&\left((\sum_{\nu=1}^{F} x_{\nu,\tau}-1)\log v_\tau - (\sum_{\nu=1}^{F} t_{\nu,i})v_\tau - \log\Gamma(\sum_{\nu=1}^{F} x_{\nu,\tau}) + (\sum_{\nu=1}^{F} x_{\nu,\tau})\log(\sum_{\nu=1}^{F} t_{\nu,i})\right)\\
&+\left(c+\log\Gamma(\sum_{\nu=1}^{F} x_{\nu,\tau}) - (\sum_{\nu=1}^{F} x_{\nu,\tau})\log(\sum_{\nu=1}^{F} t_{\nu,i}) + \sum_{\nu=1}^{F} x_{\nu,\tau}\log t_{\nu,i} - \log\Gamma(x_{\nu,\tau}+1)\right)\\
=&c+\hat{g}(x_{1:F,\tau},t_{1:F,i}) + \log\mathcal{G}(v_\tau;(\sum_{\nu=1}^{F} x_{\nu,\tau}),(\sum_{\nu=1}^{F} t_{\nu,i}))\\
=&c'+\log\mathcal{G}(v_\tau;a',b')
\end{aligned}
\tag{A.9}
$$

where

$$
\begin{aligned}
a' =&\sum_{\nu=1}^{F} x_{\nu,\tau}\\
b' =&\sum_{\nu=1}^{F} t_{\nu,i}\\
c' =&c+\hat{g}(x_{1:F,\tau},t_{1:F,i})
\end{aligned}
\tag{A.10}
$$

and

$$
\hat{g}(.) = \log\Gamma(\sum_{\nu=1}^{F} x_{\nu,\tau}) - (\sum_{\nu=1}^{F} x_{\nu,\tau})\log(\sum_{\nu=1}^{F} t_{\nu,i}) + \sum_{\nu=1}^{F} x_{\nu,\tau}\log t_{\nu,i} - \log\Gamma(x_{\nu,\tau}+1).
$$

REFERENCES

Alpaydin, E., 2004, *Introduction to Machine Learning (Adaptive Computation and Machine Learning)*, The MIT Press.

Barber, D. and A. T. Cemgil, 2010, "Graphical models for time-series", *IEEE Signal Processing Magazine Special Issue on Graphical Models in Signal Processing (to be published)*.

Bertin, N., C. Févotte, and R. Badeau, 2009, "A tempering approach for Itakura-Saito non-negative matrix factorization. With application to music transcription", *ICASSP'09*.

Cappé, O., E. Moulines, and T. Ryden, 2005, *Inference in Hidden Markov Models (Springer Series in Statistics)*, Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Cemgil, A. T., 2004, *Bayesian Music Transcription*, Ph.D. thesis, Radboud University of Nijmegen.

Cemgil, A. T., 2006, "Sequential inference for Factorial Changepoint Models", *Nonlinear Statistical Signal Processing Workshop*, IEEE, Cambridge, UK.

Cemgil, A. T., 2009, "Bayesian Inference in Non-negative Matrix Factorisation Models", *Computational Intelligence and Neuroscience*, , No. Article ID 785152.

Cemgil, A. T., H. J. Kappen, and D. Barber, 2006, "A generative model for music transcription", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 14, No. 2, pp. 679–694.

Christensen, M. G. and A. Jakobsson, 2009, *Multi-Pitch Estimation*, Morgan & Claypool Publishers.

Cont, A., 2006, "Realtime multiple pitch observation using sparse non-negative con-

straints", *in International Conference on Music Information Retrieval*.

Dannenberg, R. B., 1993, "A brief survey of music representation issues, techniques, and systems", *Computer Music Journal*, Vol. 17, pp. 20 – 30.

de Cheveigné, A. and H. Kawahara, 2002, "YIN, a fundamental frequency estimator for speech and music", *J. Acoust. Soc. Am.*, Vol. 111, pp. 1917–1930.

Excell, D., A. T. Cemgil, and W. J. Fitzgerald, 2007, "Generative Model for Human Motion Recognition", *Proc. 5th International Symposium on Image and Signal Processing and Analysis ISPA 2007*, pp. 423–428.

Godsill, S., A. Cemgil, C. Fevotte, and P. Wolfe, 2007, "Bayesian computational methods for sparse audio and music processing", *15th European Signal Processing Conference*.

Harvey, A. C., S. J. Koopman, and N. Shephard (eds.), 2004, *State Space and Unobserved Component Models: Theory and Applications. Proceedings of a Conference in Honour of James Durbin*, Cambridge University Press, Cambridge.

Jordan, M. I., 2004, "Graphical models", *Statist. Sci.*, Vol. 19, p. 140155.

Jylhä, A. and C. Erkut, 2008, "Inferring the hand configuration from hand clapping sounds", *11th Intl. Conf. Digital Audio Effects (DAFx-08)*.

Kashino, K., K. Nakadai, T. Kinoshita, and H. Tanaka, 1998, "Application of Bayesian Probability Network to Music Scene Analysis", .

Klapuri, A., 2008, "Multipitch Analysis of Polyphonic Music and Speech Signals Using an Auditory Model", *IEEE Transactions on Audio, Speech & Language Processing*, Vol. 16, No. 2, pp. 255–266.

Klapuri, A. and M. Davy (eds.), 2006, *Signal Processing Methods for Music Transcription*, Springer, New York.

Orio, N. and M. S. Sette, 2003, "An HMM-based pitch tracker for audio queries", *ISMIR*.

Parsons, S., 1998, "An introduction to Bayesian networks by Finn V. Jensen, UCL Press, 1996, 29.95, pp 178, ISBN 1-85728-332-5.", *Knowl. Eng. Rev.*, Vol. 13, No. 2, pp. 201–208.

Raphael, C., 2002, "Automatic Transcription of Piano Music", *ISMIR*.

Roads, C., 1996, *The Computer Music Tutorial*, MIT Press, Cambridge, MA, USA.

Vincent, E., N. Bertin, and R. Badeau, 2008, "Harmonic and inharmonic Nonnegative Matrix Factorization for Polyphonic Pitch transcription", *ICASSP*.

Virtanen, T., A. T. Cemgil, and S. Godsill, 2008, "Bayesian extensions to non-negative matrix factorisation for audio signal modelling", *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2008*, pp. 1825–1828.

Wainwright, M. J. and M. I. Jordan, 2008, *Graphical Models, Exponential Families, and Variational Inference*, Now Publishers Inc., Hanover, MA, USA.

West, M. and J. Harrison, 1997, *Bayesian forecasting and dynamic models (2nd ed.)*, Springer-Verlag New York, Inc., New York, NY, USA.

Whiteley, N., A. T. Cemgil, and S. Godsill, 2007, "Sequential Inference of Rhythmic Structure in Musical Audio", *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2007*, Vol. 4, pp. IV–1321–IV–1324.

Whiteley, N., A. T. Cemgil, and S. J. Godsill, 2006, "Bayesian Modelling of Temporal Structure in Musical Audio", *Proceedings of International Conference on Music Information Retrieval*.